ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Instituto de Ciencias Matemáticas

Ingeniería en Estadística Informática

"DISEÑO E IMPLEMENTACIÓN DE UN TUTORIAL INTERACTIVO DE SQL SERVER"

TESIS DE GRADO

Previa a la obtención del Título de:

INGENIERO EN ESTADÍSTICA INFORMÁTICA

Presentada por:

David Octavio Rugel González

GUAYAQUIL - ECUADOR

AÑO

2005

AGRADECIMIENTO

Al Ing. Juan Alvarado que con sus conocimientos me ha guiado en la elaboración de esta tesis para poder convertirme en un profesional.

A los ingenieros Dalton Noboa y Christian Rochina, amigos de muchos años que me han apoyado con sus conocimientos y experiencia.

A mis hermanos Yitzak, Galo, Roberto, Franklin, Noe, Ronny y muchos otros que son mi familia y en todo momento me han brindado su apoyo.

A una persona muy especial que Dios puso en mi vida y me ha brindado su apoyo incondicional en todo momento “Angélica Prieto”

A todos mis amigos(as) de la universidad que en todo momento me demostraron una amistad sincera.

DEDICATORIA

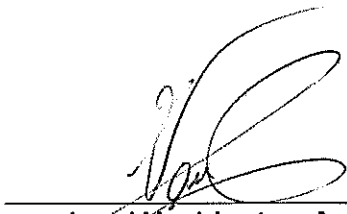
Dedico este trabajo a Dios por guiarme por el camino del bien, por iluminarme y cuidarme todos los días de mi vida y haberme ayudado y enseñado muchas cosas que me hacen sentir un ser mejor.

A mis padres María Magdalena González Duque y Lorenzo David Rugel Tutivén que siempre me han inculcado a ser una persona de bien, a ser trabajador, dedicarme a mis estudios para llegar a ser una persona de éxito en la vida.

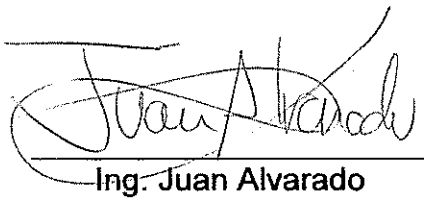
A mis hermanos: Rosa y Andrés que son seres muy lindos que siempre me han apoyado y ayudado en mis tareas diarias.

A mis abuelos, tíos, primos y demás familia que siempre me han aconsejado y me han demostrado la importancia de los valores morales.

TRIBUNAL DE GRADUACIÓN

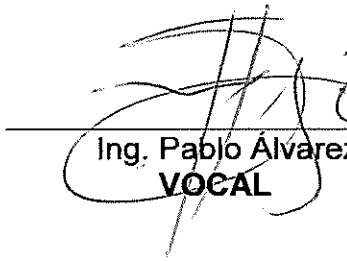


Ing. Washington Armas
PRESIDENTE DEL TRIBUNAL



Ing. Juan Alvarado
DIRECTOR DE TESIS

CIB -ESPOL



Ing. Pablo Álvarez
VOCAL

Ing. Soraya Solís
VOCAL

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de esta Tesis de Grado, me corresponde exclusivamente; y el patrimonio intelectual de la misma a la Escuela Superior Politécnica del Litoral".

CIB -ESPOL

David Octavio Rugel González

RESUMEN

El objetivo de este tutorial es permitir a sus usuarios auto evaluarse en el conocimiento y aprendizaje del manejo de consultas de bases de datos. Los tipos de consultas que se han considerado para la evaluación son de dos tipos. El primer tipo de consultas es simple, con el uso de las instrucciones básicas que son el SELECT, FROM y WHERE que incluyen el uso de una sola tabla con un valor específico del campo que se desea en la consulta. El otro tipo de consulta es de un nivel intermedio, en la cual se considera las instrucciones básicas SELECT, FROM y WHERE añadiendo el uso de dos tablas relacionadas en las que se considera valores específicos de campos y el join necesario de las dos tablas relacionadas.

Para la generación de la consulta que debe evaluar el usuario he considerado tres modelos de bases de datos, que son muy utilizadas en la vida cotidiana. El primero es un modelo de facturación simple que puede ser fácilmente acoplado a cualquier empresa tipo comercial para control de entradas y salidas e incluso implementar un inventario. El segundo es un modelo de control de nóminas de empleados que puede ser utilizado para controlar las labores que realiza cada empleado en su área respectiva, pagos de sueldos a empleados, jerarquías entre empleados, etc. El último modelo utilizado para la generación de la consulta es el de un sistema

académico que puede ser modificado por cualquier institución educativa dependiendo de sus necesidades para controlar el registro de sus alumnos en las materias que se dicten, así como los profesores que dicten las materias, cupos por paralelos, notas de alumnos por período de estudio, entre otras cosas.

En el primer capítulo de esta tesis hago mención a los sistemas multimedia, historia, inicios, su evolución, el uso social, sistemas de información y documentación. Además se trata acerca de los sistemas multimedia en la educación superior, así como también las tecnologías en los procesos educativos y una introducción a los tutoriales.

En el segundo capítulo se trata acerca del modelo relacional de bases de datos, su estructura y definición de de términos importantes.

En el tercer capítulo se presenta el tutorial interactivo desarrollado para la evaluación de consultas, presentación de su diseño con todas las especificaciones necesarias, implementación, pruebas y su respectivo manual de usuario y de instalación.

Finalmente tenemos las conclusiones y recomendaciones.

Indice

Pag.

INDICE GENERAL

CAPÍTULO 2

Modelo relacional de Bases de Datos y Lenguaje de Consultas

CAPÍTULO 3

Diseño e implementación del tutorial interactivo

INDICE DE TABLAS

INDICE DE GRÁFICOS

Abreviaturas

MIT	(Siglas en inglés: Massachussets Institute of Technology)
SDMS	Sistema de gestión especial de los datos
Hiper	de gran magnitud (muy grande)
TI	Tutoriales inteligentes
SGBD	Sistema generador de base de datos
TICSS	Tutorial interactivo de consultas de SQL Server
RDBMS	Sistema administrador de bases de datos Relacionales
IIS	(Siglas en inglés: Internet Information Server)
ODBC	(Siglas en inglés: Object Data Base Connection)
SQL	(Siglas en ingles: Standar Query Language)

CAPITULO 1

1.1- Introducción a los Sistemas Multimedia

El rápido desarrollo de la tecnología y de la informática está proporcionando herramientas que revolucionan en todos los campos de la ciencia.

Los sistemas interactivos multimedia se están integrando en nuestro entorno y cada vez hay más productos y es por eso que nos encontramos en un proceso de transformación social, que es consecuencia de tres pilares básicos iniciados a finales de los años 60 y principios de los 70. Estos pilares son: **la revolución tecnológica** (basada en el auge y desarrollo de las tecnologías de la información y la comunicación), la formación de la **economía global mundial** y el **cambio cultural** en la sociedad. La tecnología multimedia está evolucionando y es así como poco a poco ha entrado con más fuerza en la sociedad. Por ejemplo: al trabajo, a la cultura y por supuesto a la educación.

Los sistemas multimedia ofrecen combinaciones de texto, audio y vídeo en un mismo documento que son coordinadas (producidas, controladas y mostradas) por el computador. Esta integración de sonido, texto e imágenes de alta calidad (gráfico, animaciones y vídeo) en el computador producen un trabajo de alta calidad, tal que el impacto del gráfico se realza con la

integración del audio y el texto, lo que hace que su uso cada vez parezca más ilimitado.

La principal ventaja de los sistemas interactivos multimedia es que permite al usuario desplazarse, adelantarse, consultar y repetir los conceptos que se le van presentando y que más le han parecido interesantes. La diferencia básica de la tecnología multimedia con otras tecnologías es que no existen limitaciones con respecto al tiempo, imágenes y máquinas; además que esta provee un acceso mucho más amplio a la información, la misma que puede presentarse de diversas formas, dependiendo de los requerimientos y tipo de sistema multimedia, con lo cual los usuarios tienen la libertad de acceder a cualquier medio del sistema de acuerdo a sus prioridades o necesidades con diferentes perspectivas.

1.1.1.- Evolución Histórica de los Sistemas Multimedia

Desde el comienzo de la era informática los terminales de salida de información han ido mejorando a través del tiempo. Al principio sólo comprendían una simple impresora, luego fueron apareciendo las pantallas de visualización en las que los datos aparecían con mucha mayor rapidez que en una impresora. Pero estos primeros sistemas de visualización presentaban numerosos inconvenientes: el más grave consistía en el largo y engorroso trabajo de examinar un gran número de datos expresados en

forma de palabras y frases, es decir, codificados en caracteres alfabéticos y numéricos en continua sucesión temporal. Este sistema no era capaz de aprovechar la extraordinaria capacidad que tiene la visión del ser humano para localizar rápidamente objetos o situaciones complejas. La realización de un sistema de imágenes interactivas requería dos progresos: la mejora de las pantallas y la disponibilidad de memorias electrónicas de gran capacidad.

Hoy en día se puede considerar que los verdaderos multimedia tienen su comienzo en 1978 cuando el Architecture Machine Group del Massachussets Institute of Technology presentó el primer sistema combinado de ordenadores y videodiscos. El grupo de arquitectura de máquina del MIT diseñó lo que denominan SDMS (sistema de gestión especial de los datos), que era un sistema basado en explorar las posibilidades de las imágenes como representación espacial para acceder a la información almacenada en bases de datos electrónicas. Los datos se buscaban en un gráfico representado visualmente en pantalla, en vez de solicitarlos mediante una serie de órdenes escritas verbales y numéricas. El sistema partía de la orden especial del ser humano para localizar rápidamente y de modo preciso los objetos en el espacio. El SDMS constituyó una alternativa al acceso habitual a los datos en una base simbólica, pero en ningún momento se planteó la utilización de la imagen interactiva como un sustituto, sino como un complemento del uso de los teclados. Como consecuencia de las

investigaciones del Architecture Machine Group se desarrolló una serie de aplicaciones, siendo la más popular el <plano-película> de Aspen. Para realizar el que sería el primer multimedia se grabaron en soporte cinematográfico las calles de la ciudad de Aspen (Colorado), filmando cada calle en las dos direcciones y con una cadencia de un fotograma por metro real de la calle. Al montar en un videodisco los segmentos de calle rectos y en otro videodisco las curvas, el ordenador permitía la sensación de estar conduciendo. Se podía mirar por la ventanilla, parar delante de un edificio, entrar, o volar en helicóptero sobre mapas “reales” e ir dejando una huella sobre el camino recorrido que permitiera el regreso. La meta a conseguir era la interacción total, en tiempo real, entre el usuario y el sistema de tratamiento de los datos, como si éste se tratara de un auténtico interlocutor; pero un interlocutor sumiso y obediente a las instrucciones que el usuario le suministre por medio de sus dedos, de sus ojos, o de su voz. En EEUU, en 1979, se producen las primeras aplicaciones comerciales de video interactivo: General Motors instaló 12.000 unidades de videodisco industrial en su red de distribuidores. Y en 1980, Pioneer sacó al mercado su primer reproductor LaserDisc de tipo doméstico.

A principios de los años 80 se inició el desarrollo de equipos para almacenar información en formato óptico, este tipo de tecnología supuso la posibilidad de almacenar una mayor información en un espacio menor, y por lo tanto un

paso imprescindible para el almacenamiento de imágenes en soporte informático. Al soporte desarrollado se denominó videodisco y aportaba una importante característica para el desarrollo posterior de los multimedia, y es que su lector era fácilmente controlable por medio de un computador.

1.1.2.- El uso social de los multimedia

Sería un poco apresurado realizar una clasificación acerca de los géneros que abarcan los multimedia. En las que se presentan a través del tiempo, se atiende principalmente a los usos sociales de los contenidos y también se observa una cierta tendencia a tomar como referencia los géneros televisivos. Es por eso que en este trabajo nos limitaremos a mencionar los principales campos de aplicación para los sistemas multimedia, que se pueden encontrar entre los siguientes:

- Sistemas de información y documentación
- Educación y formación de personal
- Entretenimiento
- Publicidad

1.1.2.1.- Sistemas de información y documentación

La gran capacidad de información que pueden contener los multimedia y su rápido y fácil acceso los convierte en medios adecuados para almacenar contenidos de tipo informativo y documental. Esto lo podemos notar en los

terminales de información utilizados en museos y exposiciones (en los que también se puede unir lo didácticos a lo informativo). En otro lugar donde podemos analizar los multimedia son los puntos de información para turistas, planos-guía interactivos de ciudades o lugares a visitar, etc. "http://www.louvre.or.jp/louvre/QTVR/anglais/index.htm"

Con el transcurso del tiempo podemos inferir que las enciclopedias son probablemente una de las aplicaciones más prometedoras de los sistemas multimedia, por la posibilidad de aprovechar los recursos interactivos de este soporte, por la gran cantidad de información que son capaces de contener, por su flexibilidad en la forma y modo de acceso a los contenidos. Y sobre todo, por la incorporación de sonido e imágenes dinámicas que dan como resultado una facilidad comunicativa imposible de alcanzar en las obras convencionales de papel. "http://encarta.msn.com/"

1.1.2.2.- Educación y formación de personal

Cuando un programa es interactivo, el receptor se ve obligado a participar si quiere avanzar, es necesario prestar atención y responder los requerimientos del programa. De aquí se deduce el especial interés que los programas multimedia interactivos pueden tener en el campo educativo. http://www.clase.net.

Por otra parte los programas multimedia interactivos permiten la simulación de situaciones reales que los capacita para utilizarlos en un tipo de aprendizaje próximo al que se realiza en dichas situaciones. Y con la ventaja de un costo mucho menor.

Una de las primeras aplicaciones de este tipo fue la formación de pilotos de aeronaves mediante simuladores especiales diseñados para reflejar lo más real que se pudiera las situaciones que se pueden dar al pilotar un avión; y hoy en día cualquiera puede instalar un programa de simulación de vuelo en su computador.

1.1.2.3.- Entretenimiento

Los videojuegos han llegado a ser calificados por algunos como “cine interactivo”, pero en realidad tienen un uso social muy específico y actualmente constituyen un mercado de una rentabilidad superior a la de la industria del cine. De todas las aplicaciones informáticas que han ido surgiendo en los últimos años en torno a la imagen y los medios audiovisuales, los videojuegos constituyen el primer éxito comercial. Los juegos de inmersión en un espacio virtual podemos encontrarlos en parques de atracciones o, los más modestos, en salas de juegos recreativos. En esencia son simuladores de algún tipo de experiencia: viaje en un avión a reacción que rompe la barrera del sonido, viaje a la Luna en el trasbordador

espacial, viaje a Marte de diez minutos, expedición al lago Ness, mina del diablo con piscinas llenas de ácido corrosivo, túneles de cal viva, barriles de pólvora que estallan, etc.

1.1.2.4.- Publicidad

Es frecuente el uso de los sistemas multimedia interactivos con fines publicitarios. Las primeras aplicaciones destacables de los multimedia a la publicidad han sido las del sector financiero. En EEUU, más del 90% de los Bancos han introducido o tienen planes de implantar sistemas de información interactiva y autoservicio para el cliente. De forma similar a la instalación de cajeros automáticos, se instalan terminales, frecuentemente multimedia, destinados a ofrecer todo tipo de información financiera a los potenciales clientes. También se pueden establecer puntos de información comercial en ferias y superficies de venta. “http://www.bancomer.com.mx” “http://www.banamex.com.mx”

1.2.- Los Sistemas Multimedia en la Educación Superior

La aplicación de la tecnología multimedia en la educación superior apunta actualmente la necesidad de un replanteamiento teórico de la investigación y evaluación de las nuevas tecnologías en la práctica educativa. La ausencia de una perspectiva comunicacional y el dominio de la racionalidad

instrumental en la introducción de software informático deben ser contrarrestados por una reflexión socio-pedagógica, hoy prácticamente inexistente, sobre los discursos e ideologías de la información y la comunicación, de la que participan las diferentes experiencias en educación electrónica con los nuevos medios de aprendizaje.

La aplicación de los sistemas multimedia de comunicación en el contexto universitario es un problema de escritura más que de lectura. De escritura porque la cultura del hipertexto modifica las categorías y modelos de conocimiento tradicionales y de lectura, porque el nuevo sistema multimedia, cuestiona por fin la concepción informática de la comunicación, revelando el verdadero carácter interactivo de toda información y el carácter complejo, dinámico y abierto de la comunicación como espacio de construcción del conocimiento, de las identidades culturales y de la organización social.

El actual desarrollo tecnológico está obteniendo significativos cambios en la concepción del sujeto y el conocimiento de la realidad por la transformación de las tradicionales categorías del tiempo, del espacio y de las relaciones sociales. El texto de la nueva cultura universitaria debe ser el hipertexto, una nueva forma de escritura que exige habilidades distintas de competencia comunicacional para la lectura y el aprendizaje significativo de los universitarios en el ambiente de la cultura electrónica. El desarrollo de los

computadores y la tecnología multimedia ha llamado, por ello, de inmediato la atención de la comunidad universitaria, necesitada de software adecuado para hacer frente al aumento exponencial de la información y el conocimiento.

Desde prácticamente la década de los ochenta, el hipertexto y los sistemas multimedia son parte integrante, aunque marginal, del diseño de instrucción y la planificación de tecnologías educativas en muchas universidades, abriendo la puerta a una infinidad de problemas aún no suficientemente investigados. El problema de la aplicación de las tecnologías multimedia en la educación universitaria no debe limitarse a la adaptación didáctica de los equipos y programas informáticos. Conviene además tomar en consideración las implicaciones socioculturales que conlleva la ampliación del negocio de las multinacionales telemáticas en el actual proceso de explosión educativa.

Toda experiencia en educación audiovisual, o en tecnologías de la información para la educación, remite a una teoría pedagógica, y esta a su vez participa de un planteamiento político referido a un proyecto social en concreto. Conocer la naturaleza de los discursos que dejan cada una de las experiencias en nuevas tecnologías de la información y analizar las implicaciones ideológicas y prácticas del uso de los medios en el contexto histórico-cultural concreto de la educación es una tarea inevitable para una

integración crítica de los nuevos medios de aprendizaje en la enseñanza universitaria. La introducción de la tecnología multimedia en la Universidad nace asociada a un proyecto ideológico de innovación y reforma educativa específico. El desarrollo informático en las aulas, su actual crecimiento exponencial en los sistemas formales de enseñanza, se produce en un momento de transformación de la filosofía educativa según premisas clásicas del productivismo industrial. Desde la década de los setenta, los arquitectos del sistema mundial consideraban el campo educativo como la punta de lanza del desarrollo de capital humano en el proceso de transnacionalización y comercialización de mercancías. En el nuevo modelo de desarrollo educativo, el Banco Mundial exige una firme política de modernización de los sistemas nacionales de educación, al objeto de hacer coincidir los objetivos de los educadores con las necesidades de empresarios, políticos y otros grupos sociales, que demandan una transformación institucional de la educación según los parámetros del crecimiento económico. En una economía en la que el conocimiento es un valor material estratégico, la educación y la capacitación en ciencia y tecnología adquiere así un valor significativo. Hoy la formación de la fuerza de trabajo se considera un elemento muy importante y delicado de la competitividad de la economía. La educación, más aún, es calificada como un instrumento para el cambio, como una inversión social para el trabajo y el bienestar económico de las naciones e incluso como una forma de actividad productora de conocimiento y de

saber productivo. En el ámbito educativo pues se condensan gran parte de los mitos, los enredos y los líos tecnológicos que nos propone el paradigma de la sociedad electrónica. El factor tecnológico concentra los viejos y ancestrales mitos de la ideología capitalista del progreso, proyectando las visiones de un mundo integrado eficazmente en razón de la ciencia y la tecnología. En la actualidad, tres factores convergen en el desarrollo económico tardo capitalista de la sociedad informacional:

- La evolución tecnológica, que acrecienta las posibilidades técnicas de los nuevos medios de comunicación.
- La diferenciación, segmentación e individualización de los medios de consumo cultural a través de los nuevos medios y servicios informativos disponibles por los usuarios.
- Y la interpenetración de los intereses y de las estrategias del capital financiero e industrial en el campo de la comunicación y la cultura.

La atracción docente por los nuevos sistemas de enseñanza multimedia carece de una crítica pedagógica que medie en el proceso de comprensión de las nuevas máquinas de aprendizaje no como "tecnologías de la libertad", sino como sistemas técnicos susceptibles de diversos usos y prácticas sociales. La aplicación de software informático en la educación responde, fundamentalmente, a dos racionalidades bien distintas: la lógica basada en conocimientos técnicos y la perspectiva pedagógica. La mayor o menor

capacidad reflexiva de la comunidad docente, la actitud de rechazo o aceptación de los nuevos sistemas de educación y aprendizaje por los profesionales de la educación, es uno de los factores incisivos más importantes del retraso tecnológico en la educación superior. Por ello, una condición inexcusable de la investigación en Comunicación Educativa para plantearse el problema de aplicación de la tecnología multimedia es la propia reflexión y la investigación cooperativa de los profesionales de la educación, y por supuesto, también del conjunto de la comunidad universitaria sobre los problemas, retos, mitos, discursos, ideologías y contradicciones que comprende el uso de los nuevos medios de aprendizaje y de comunicación y educación en la educación superior.

Los docentes, a partir de lo citado anteriormente, tienen que enfrentar temas de discusión, tales como:

- Tendencias que promueve el nuevo sistema de comunicación en el sistema educativo a través de la creciente centralización y patriotismo de las fuentes de información y conocimiento
- Cambios geopolíticos que producen la explosión informativa hipermedia en el sistema de socialización educativa
- Formas de dependencia y qué posibilidades de autodeterminación están favoreciendo estos medios electrónicos en el sistema universitario

- Relaciones que se establecen entre la multiplicidad y la diversidad de las culturas en el proceso de globalización de las redes telemáticas
- Innovaciones para lograr una incorporación democrática y transformadora del entorno universitario con las tecnologías "hiper" del sistema económico "trans-nacional"
- Desigualdades y contradicciones que introduce los sistemas multimedia en el interior de la Universidad
- Alternativas de formalización educativa que son viables en el uso de los sistemas hipermedia más allá de la saturación y la fragmentación de las rutas textuales de adquisición de la información y el conocimiento
- Condiciones para elaborar una Teoría de la Comunicación Educativa que nos aporte un sistema de categoría y, sobre todo, con metodología de aplicación y evaluación de las experiencias con estos nuevos soportes de aprendizaje
- Criterios que deberían guiar la implementación y desarrollo multimedia en la educación
- Políticas de innovación y renovación tecnológica que debe desarrollar la Universidad, más allá de las estrategias de obsolescencia planificada de la industria electrónica

1.3.- Tecnología en los procesos educativos

Ante el desafío de una educación globalizada y la tendencia cada día más marcada hacia la internacionalización en todos los ámbitos, las universidades deben considerar la necesidad de incorporar tecnología en los procesos educativos para desarrollar nuevas formas de aprendizaje, a través del acceso a múltiples formas de interacción y fuentes de información.

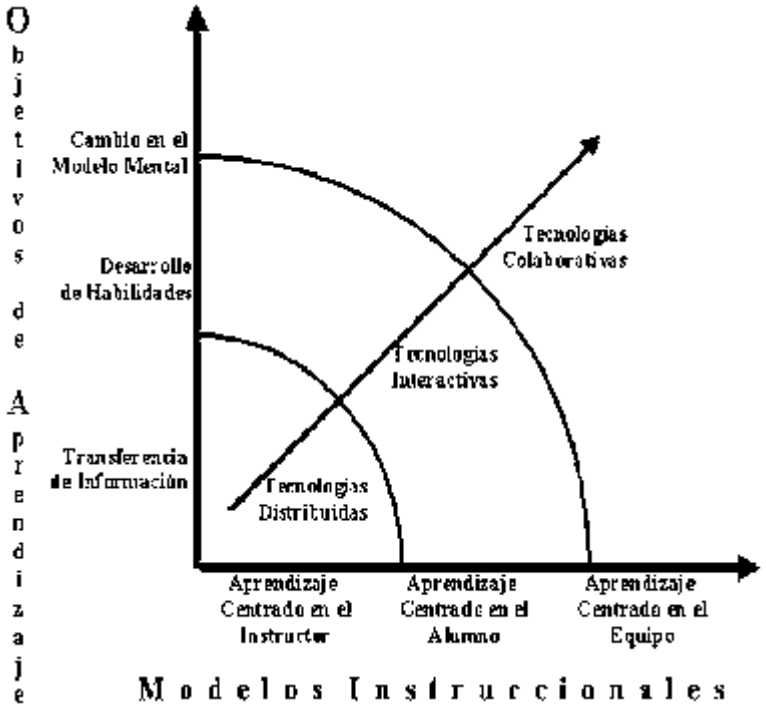
La tecnología que se puede incorporar para facilitar un modelo educativo tiene diversos componentes. Los diferentes cursos que se imparten se diseñan para que se fortalezca el nivel académico de los mismos y así asegurar la formación de profesionales competitivos a nivel internacional, para que se incorporen, en forma sistemática y planeada a actividades tendientes a desarrollar valores, actitudes y habilidades, y se añadan procesos didácticos de relevancia para el futuro desempeño del egresado.

Una vez diseñados los cursos, estos son incorporados en la plataforma computacional. El *Lotus Institute* ha sido pionero en la investigación y desarrollo sobre aprendizaje distribuido y en el diseño de soluciones y métodos tecnológicos que soporten el aprendizaje colaborativo en cualquier lugar y en cualquier momento. Sus resultados han salido al mercado a través de Lotus Education e IBM en la aplicación llamada Lotus® Learning Space™, que incorpora la riqueza del aprendizaje de grupo con la flexibilidad

del aprendizaje individual. Las características principales de esta herramienta son:

El aprendizaje se centra en el alumno, ya que los estudiantes pueden navegar, explorar y discutir la información. Los alumnos pueden trabajar en forma individual, a su propio ritmo, de manera asincrónica grupalmente. Su fuerza reside en la habilidad de soportar el aprendizaje en un ambiente distribuido. Los estudiantes colaboran en la solución de problemas, discusiones y ejercicios que producen la creación de nuevo conocimiento, que se captura y almacena. Posteriormente se utiliza para enriquecer el trabajo cooperativo del grupo. La plataforma computacional es un complemento a las actividades diarias del alumno en el salón de clase y le permite verificar y dar seguimiento al plan de trabajo del profesor. Incorpora flexibilidad al proceso de enseñanza – aprendizaje. En la siguiente gráfica se observa diferentes tipos de objetivos de aprendizaje, que van desde la transferencia de información hasta un cambio en el estado mental. Los esquemas que la gran mayoría de las instituciones educativas siguen están enfocados a transmitir información, como lo comentamos son modelos educativos centrados en la enseñanza. El siguiente nivel incorpora el desarrollo de habilidades, el cual se ve presente en algunos modelos educativos, mientras que cuando se busca un cambio en el modelo mental, estamos hablando de una transformación de la persona y de su rol en el modelo educativo.

Gráfico 1



La otra dimensión de la figura anterior está relacionada con los *Modelos instruccionales.* El aprendizaje centrado en el instructor es el que la mayoría de las instituciones educativas utiliza, donde el profesor juega el rol principal e imparte conocimiento. Cuando se lleva el centro del aprendizaje hacia el alumno, cambiamos el rol del mismo en su propio aprendizaje, y cuando centramos la responsabilidad del proceso en un grupo de alumnos y profesores, modifica el esquema y la manera de aprender. Distinguimos tres tipos de tecnologías: las tecnologías distribuidas para apoyar la transferencia de información en modelos instruccionales centrados en el profesor, como por ejemplo clases vía satélite (videoconferencias, tele secundaria, entre

otras); las tecnologías interactivas, como por ejemplo tutoriales y los sistemas de autoaprendizaje, que soportan el desarrollo de habilidades en ambientes de aprendizaje centrados en el alumno; las herramientas centradas en el equipo, son las tecnologías colaborativas, que ayudan a producir un cambio en el modelo mental, del alumno y del profesor, que soportan una instrucción centrada en el grupo de trabajo, de manera asincrónica y distribuida, entre ellas están los ambientes de administración del conocimiento, ambientes de trabajo colaborativo y los sistemas de comunicación interactiva, entre otros.

1.4.- Tutoriales

Los tutores inteligentes (TI) son programas de ordenador que utilizan técnicas procedentes del campo de la inteligencia artificial para presentar el conocimiento y llevar a cabo una interacción con el alumno. El objetivo fundamental de los tutores inteligentes es proporcionar una instrucción más adaptada, tanto en el contenido como en la forma. Todo TI debe poseer una base de conocimientos que contenga el conjunto de informaciones que el sistema ha de proporcionar y un intérprete que decida cómo y cuando aplicar el conocimiento empleado por el sistema. Son programas que no se centran únicamente en realizar tareas de tipo repetitivo, sino que el ordenador es una ayuda para realizar tareas complejas, tales como el diagnóstico o la toma de decisiones. Han sido concebidos para simular decisiones en procesos

multivariantes. Los hay de dos tipos: los no aprendientes y los aprendientes, caracterizados por retroalimentarse en función de las decisiones anteriores que ha dado el alumno. Su acción se basa en el diseño de lo que se denominan “entornos reactivos” y que actúan en base a los siguientes componentes:

- modelo bibliográfico (temas y contenidos),
- modelo de alumno (saber qué errores se cometen y por qué, para promover estrategias de aprendizaje que se anticipen a ellos),
- modelo de profesor; simula la conducta del profesor:

1. Orientando al alumno sobre la forma de resolver el problema, para que pueda ensayar alternativas de solución,
2. Formulando preguntas al alumno que le ayudan a razonar y a formular o modificar sus propios conceptos. A través de los ejercicios o juegos, el alumno descubre leyes o hechos,
3. Proporcionando tareas para evaluar respuestas y detectar concepciones erróneas.

CAPITULO 2

MODELO RELACIONAL DE BASES DE DATOS Y LENGUAJE DE CONSULTAS

2.1 Introducción al modelo relacional

Para realizar la introducción al modelo relacional se han considerado párrafos de los textos de Batini, Ceri y Navathe (1994) y de la dirección electrónica "http://www.itlp.edu.mx".

El diseño de bases de datos es el proceso por el que se determina la organización de una base de datos, incluido su estructura, contenido y las aplicaciones que se han de desarrollar.

Durante mucho tiempo, el diseño de bases de datos fue considerado una tarea para expertos: más un arte que una ciencia. Sin embargo, se ha progresado mucho en el diseño de bases de datos y éste se considera ahora una disciplina estable, con métodos y técnicas propios. Debido a la creciente aceptación de las bases de datos por parte de la industria y el gobierno en el plano comercial, y a una variedad de aplicaciones científicas y técnicas, el diseño de bases de datos desempeña un papel central en el empleo de los recursos de información en la mayoría de las organizaciones.

El diseño de bases de datos ha pasado a constituir parte de la formación general de los informáticos, en el mismo nivel que la capacidad de construir algoritmos usando un lenguaje de programación convencional. Las últimas dos décadas se han caracterizado por un fuerte crecimiento en el número e importancia de las aplicaciones de bases de datos. Las bases de datos son componentes esenciales de los sistemas de información, usadas rutinariamente en todos los computadores. El diseño de bases de datos se ha convertido en una actividad popular, desarrollada no sólo por profesionales sino también por no especialistas. A finales de la década de 1960, cuando las bases de datos entraron por primera vez en el mercado del software, los diseñadores de bases de datos actuaban como artesanos, con herramientas muy primitivas: diagramas de bloques y estructuras de registros que eran los formatos comunes para las especificaciones. Además el diseño de bases de datos se confundía frecuentemente con la implantación de las bases de datos. Hoy en día esta situación ha cambiado: los métodos y modelos de diseño de bases de datos han evolucionado paralelamente con el progreso de la tecnología en los sistemas de bases de datos. Hemos incursionado en la era de los sistemas relacionales de bases de datos, que ofrecen poderosos lenguajes de consulta, herramientas para el desarrollo de aplicaciones e interfaces amables para los usuarios. La tecnología de bases de datos cuenta ya con un marco teórico, que incluye la teoría relacional de datos, procesamiento y optimización de consultas, control de concurrencia, gestión de transacciones y recuperación, etc. A

través del tiempo la tecnología de bases de datos ha dado grandes pasos, es así que se han desarrollado metodologías y técnicas de diseño. Se ha alcanzado un consenso, por ejemplo, sobre la descomposición del proceso de diseño en fases, sobre los principales objetivos de cada fase y sobre las técnicas para conseguir estos objetivos.

Desafortunadamente, las metodologías de diseño de bases de datos no son muy populares; la mayoría de las organizaciones y de los diseñadores individuales confía muy poco en las metodologías para llevar a cabo el diseño y esto se considera, con frecuencia, una de las principales causas de fracaso en el desarrollo de los sistemas de información. Debido a la falta de enfoques estructurados para el diseño de bases de datos, a menudo se subestiman el tiempo o los recursos necesarios para un proyecto de bases de datos, las bases de datos son inadecuadas o ineficientes en relación a las demandas de la aplicación, la documentación es limitada y el mantenimiento es difícil. Muchos de estos problemas se deben a la falta de una claridad que permita entender la naturaleza exacta de los datos, a un nivel conceptual y abstracto. En muchos casos, los datos se describen desde el comienzo del proyecto en términos de las estructuras finales de almacenamiento; no se da peso a un entendimiento de las propiedades estructurales de los datos que sea independiente de los detalles de la realización.

2.2 Estructura del modelo relacional

El diseño de una base de datos es un proceso complejo que abarca decisiones a muy distintos niveles. La complejidad se controla mejor si se descompone el problema en subproblemas y se resuelve cada uno de estos subproblemas independientemente, utilizando técnicas específicas. Así, el diseño de una base de datos se descompone en diseño conceptual, diseño lógico y diseño físico. El diseño conceptual parte de las especificaciones de requisitos de usuario y su resultado es el esquema conceptual de la base de datos.

Un esquema conceptual es una descripción de alto nivel de la estructura de la base de datos, independientemente del sistema implementado para la base de datos, que se vaya a utilizar para manipularla. Un modelo conceptual es un lenguaje que se utiliza para describir esquemas conceptuales. El objetivo del diseño conceptual es describir el contenido de información de la base de datos y no las estructuras de almacenamiento que se necesitarán para manejar esta información.

El diseño lógico parte del esquema conceptual y da como resultado un esquema lógico. Un esquema lógico es una descripción de la estructura de la base de datos en términos de las estructuras de datos que puede procesar un tipo de sistema implementado para el manejo de la misma. Un modelo lógico es un lenguaje usado para especificar esquemas lógicos

(modelo relacional, modelo de red, etc.). El diseño lógico depende del tipo de sistema implementado para el manejo de la base de datos que se vaya a utilizar, más no depende del producto concreto.

El diseño físico parte del esquema lógico y da como resultado un esquema físico. Un esquema físico es una descripción de la implementación de una base de datos en memoria secundaria: las estructuras de almacenamiento y los métodos utilizados para tener un acceso eficiente a los datos. Por ello, el diseño físico depende también del sistema implementado para el manejo de la base de datos concreto y el esquema físico se expresa mediante su lenguaje de definición de datos.

2.2.1 Metodología del diseño conceptual

El primer paso en el diseño de una base de datos es la producción del esquema conceptual. Normalmente, se construyen varios esquemas conceptuales, cada uno para representar las distintas visiones que los usuarios tienen de la información. Cada una de estas visiones suelen corresponder a las diferentes áreas funcionales de la empresa como, por ejemplo, producción, ventas, recursos humanos, etc. Estas visiones de la información, denominadas *vistas*, se pueden identificar de varias formas.

Una opción consiste en examinar los diagramas de flujo de datos, que se pueden haber producido previamente, para identificar cada una de las áreas funcionales. La otra opción consiste en entrevistar a los usuarios, examinar

los procedimientos, los informes y los formularios, y también observar el funcionamiento de la empresa. A los esquemas conceptuales correspondientes a cada vista de usuario se les denomina *esquemas conceptuales locales*. Cada uno de estos esquemas se compone de entidades, relaciones, atributos, dominios de atributos e identificadores. El esquema conceptual también tendrá una documentación, que se irá produciendo durante su desarrollo. Las tareas a realizar en el diseño conceptual son las siguientes:

1. Identificar las entidades.
2. Identificar las relaciones.
3. Identificar los atributos y asociarlos a entidades y relaciones.
4. Determinar los dominios de los atributos.
5. Determinar los identificadores.
6. Determinar las jerarquías de generalización (si las hay).
7. Dibujar el diagrama entidad-relación.
8. Revisar el esquema conceptual local con el usuario.

Tarea 1: Identificar las entidades

En primer lugar hay que definir los principales objetos que interesan al usuario. Estos objetos serán las entidades. Una forma de identificar las entidades es examinar las especificaciones de requisitos de usuario. En estas especificaciones se buscan los nombres (por ejemplo: número de empleado, nombre de empleado, número de inmueble, dirección del

inmueble, alquiler, número de habitaciones). También se buscan objetos importantes como personas, lugares o conceptos de interés, excluyendo aquellos nombres que sólo son propiedades de otros objetos. Por ejemplo, se pueden agrupar el número de empleado y el nombre de empleado en una entidad denominada *empleado*, y agrupar número de inmueble, dirección del inmueble, alquiler y número de habitaciones en otra entidad denominada *inmueble*. Otra forma de identificar las entidades es buscar aquellos objetos que existen por sí mismos. Por ejemplo, *empleado* es una entidad porque los empleados existen, sepamos o no sus nombres, direcciones y teléfonos. Siempre que sea posible, el usuario debe colaborar en la identificación de las entidades. A veces, es difícil identificar las entidades por la forma en que aparecen en las especificaciones de requisitos. Los usuarios, a veces, hablan utilizando ejemplos o analogías. En lugar de hablar de empleados en general, hablan de personas concretas, o bien, hablan de los puestos que ocupan esas personas. No siempre es obvio saber si un objeto es una entidad, una relación o un atributo. Los diseñadores de bases de datos deben tener una visión selectiva y clasificar las cosas que observan dentro del contexto de la empresa u organización. A partir de unas especificaciones de usuario es posible que no se pueda deducir un conjunto único de entidades, pero después de varias iteraciones del proceso de análisis, se llegará a obtener un conjunto de entidades que sean adecuadas para el sistema que se ha de construir. Conforme se van identificando las entidades, se les dan nombres que tengan un significado y que sean obvias para el

usuario. Los nombres de las entidades y sus descripciones se anotan en el diccionario de datos. Cuando sea posible, se debe anotar también el número aproximado de ocurrencias de cada entidad. Si una entidad se conoce por varios nombres, éstos se deben anotar en el diccionario de datos como alias o sinónimos.

Tarea 2: Identificar las relaciones

Una vez definidas las entidades, se deben definir las relaciones existentes entre ellas. Del mismo modo que para identificar las entidades se buscaban nombres en las especificaciones de requisitos, para identificar las relaciones se suelen buscar las expresiones verbales (por ejemplo: oficina tiene empleados, empleado gestiona inmueble, cliente visita inmueble). Si las especificaciones de requisitos reflejan estas relaciones es porque son importantes para la empresa y, por lo tanto, se deben reflejar en el esquema conceptual. Pero sólo interesan las relaciones que son necesarias. La mayoría de las relaciones son binarias (entre dos entidades), pero no hay que olvidar que también puede haber relaciones en las que participen más de dos entidades, así como relaciones recursivas. Es muy importante repasar las especificaciones para comprobar que todas las relaciones, explícitas o implícitas, se han encontrado. Si se tienen pocas entidades, se puede comprobar por parejas si hay alguna relación entre ellas. De todos modos, las relaciones que no se identifican ahora se suelen encontrar cuando se valida el esquema con las transacciones que debe soportar. Una

vez identificadas todas las relaciones, hay que determinar la cardinalidad mínima y máxima con la que participa cada entidad en cada una de ellas. De este modo, el esquema representa de un modo más explícito la semántica de las relaciones. La cardinalidad es un tipo de restricción que se utiliza para comprobar y mantener la calidad de los datos. Estas restricciones son aserciones sobre las entidades que se pueden aplicar cuando se actualiza la base de datos para determinar si las actualizaciones violan o no las reglas establecidas sobre la semántica de los datos. Conforme se van identificando las relaciones, se les van asignando nombres que tengan significado para el usuario. En el diccionario de datos se anotan los nombres de las relaciones, su descripción y las cardinalidades con las que participan las entidades en ellas.

Tarea 3: Identificar los atributos y asociarlos a entidades y Relaciones

Al igual que con las entidades, se buscan nombres en las especificaciones de requisitos. Son atributos los nombres que identifican propiedades, cualidades, identificadores o características de entidades o relaciones. Lo más sencillo es preguntarse, para cada entidad y cada relación, ¿qué información se quiere saber de..? La respuesta a esta pregunta se debe encontrar en las especificaciones de requisitos. Pero, en ocasiones, será necesario preguntar a los usuarios para que aclaren los requisitos. Desgraciadamente, los usuarios pueden dar respuestas a esta pregunta que

también contengan otros conceptos, por lo que hay que considerar sus respuestas con mucho cuidado. Al identificar los atributos, hay que tener en cuenta si son simples o compuestos. Por ejemplo, el atributo *dirección* puede ser simple, teniendo la dirección completa como un solo valor: `Av. Orellana 45, Guayaquil; o puede ser un atributo compuesto, formado por la *calle* (`Av. Orellana'), el *número* (`45') y la *población* (`Guayaquil'). El escoger entre atributo simple o compuesto depende de los requisitos del usuario. Si el usuario no necesita acceder a cada uno de los componentes de la dirección por separado, se puede representar como un atributo simple. Pero si el usuario quiere acceder a los componentes de forma individual, entonces se debe representar como un atributo compuesto. También se deben identificar los atributos derivados o calculados, que son aquellos cuyo valor se puede calcular a partir de los valores de otros atributos. Por ejemplo, el número de empleados de cada oficina, la edad de los empleados o el número de inmuebles que gestiona cada empleado. Algunos diseñadores no representan los atributos derivados en los esquemas conceptuales. Si se hace, se debe indicar claramente que el atributo es derivado y a partir de qué atributos se obtiene su valor. Donde hay que considerar los atributos derivados es en el diseño físico. Cuando se están identificando los atributos, se puede descubrir alguna entidad que no se ha identificado previamente, por lo que hay que volver al principio introduciendo esta entidad y viendo si se relaciona con otras entidades. Es muy útil elaborar una lista de atributos e ir eliminándolos de la lista conforme se vayan asociando a una entidad o

relación. De este modo, uno se puede asegurar de que cada atributo se asocia a una sola entidad o relación, y que cuando la lista se ha acabado, se han asociado todos los atributos. Hay que tener mucho cuidado cuando parece que un mismo atributo se debe asociar a varias entidades. Esto puede ser por una de las siguientes causas:

- Se han identificado varias entidades, como *director*, *supervisor* y *administrativo*, cuando, de hecho, pueden representarse como una sola entidad denominada *empleado*. En este caso, se puede escoger entre introducir una jerarquía de generalización, o dejar las entidades que representan cada uno de los puestos de empleado.
- Se ha identificado una relación entre entidades. En este caso, se debe asociar el atributo a una sola de las entidades y hay que asegurarse de que la relación ya se había identificado previamente. Si no es así, se debe actualizar la documentación para recoger la nueva relación.

Conforme se van identificando los atributos, se les asignan nombres que tengan significado para el usuario. De cada atributo se debe anotar la siguiente información:

- Nombre y descripción del atributo.
- Alias o sinónimos por los que se conoce al atributo.
- Tipo de dato y longitud.

- Valores por defecto del atributo (si se especifican).
- Si el atributo siempre va a tener un valor (si admite o no nulos).
- Si el atributo es compuesto y, en su caso, qué atributos simples lo forman.
- Si el atributo es derivado y, en su caso, cómo se calcula su valor.
- Si el atributo es multievaluado.

Tarea 4: Determinar los dominios de los atributos

El dominio de un atributo es el conjunto de valores que puede tomar el atributo. Por ejemplo el dominio de los números de oficina son las tiras de hasta tres caracteres en donde el primero es una letra y el siguiente o los dos siguientes son dígitos en el rango de 1 a 99; el dominio de los números de teléfono y los números de fax son las tiras de 9 dígitos. Un esquema conceptual está completo si incluye los dominios de cada atributo: los valores permitidos para cada atributo, su tamaño y su formato. También se puede incluir información adicional sobre los dominios como, por ejemplo, las operaciones que se pueden realizar sobre cada atributo, qué atributos pueden compararse entre sí o qué atributos pueden combinarse con otros. Aunque sería muy interesante que el sistema final respetara todas estas indicaciones sobre los dominios, esto es todavía una línea abierta de investigación. Toda la información sobre los dominios se debe anotar también en el diccionario de datos.

Tarea 5: Determinar los identificadores

Cada entidad tiene al menos un identificador. En este paso, se trata de encontrar todos los identificadores de cada una de las entidades. Los identificadores pueden ser simples o compuestos. De cada entidad se escogerá uno de los identificadores como clave primaria en la fase del diseño lógico. Cuando se determinan los identificadores es fácil darse cuenta de si una entidad es fuerte o débil. Si una entidad tiene al menos un identificador, es *fuerte* (otras denominaciones son *padre*, *propietaria* o *dominante*). Si una entidad no tiene atributos que le sirvan de identificador, es *débil* (otras denominaciones son *hijo*, *dependiente* o *subordinada*). Todos los identificadores de las entidades se deben anotar en el diccionario de datos.

Tarea 6: Determinar las jerarquías de generalización

En este paso hay que observar las entidades que se han identificado hasta el momento. Hay que ver si es necesario reflejar las diferencias entre distintas ocurrencias de una entidad, con lo que surgirán nuevas sub-entidades de esta entidad genérica; o bien, si hay entidades que tienen características en común y que realmente son sub-entidades de una nueva entidad genérica. En cada jerarquía hay que determinar si es total o parcial y exclusiva o superpuesta.

Tarea 7: Dibujar el diagrama entidad-relación

Una vez identificados todos los conceptos, se puede dibujar el diagrama entidad-relación correspondiente a una de las vistas de los usuarios. Se obtiene así un esquema conceptual local.

Tarea 8: Revisar el esquema conceptual local con el usuario

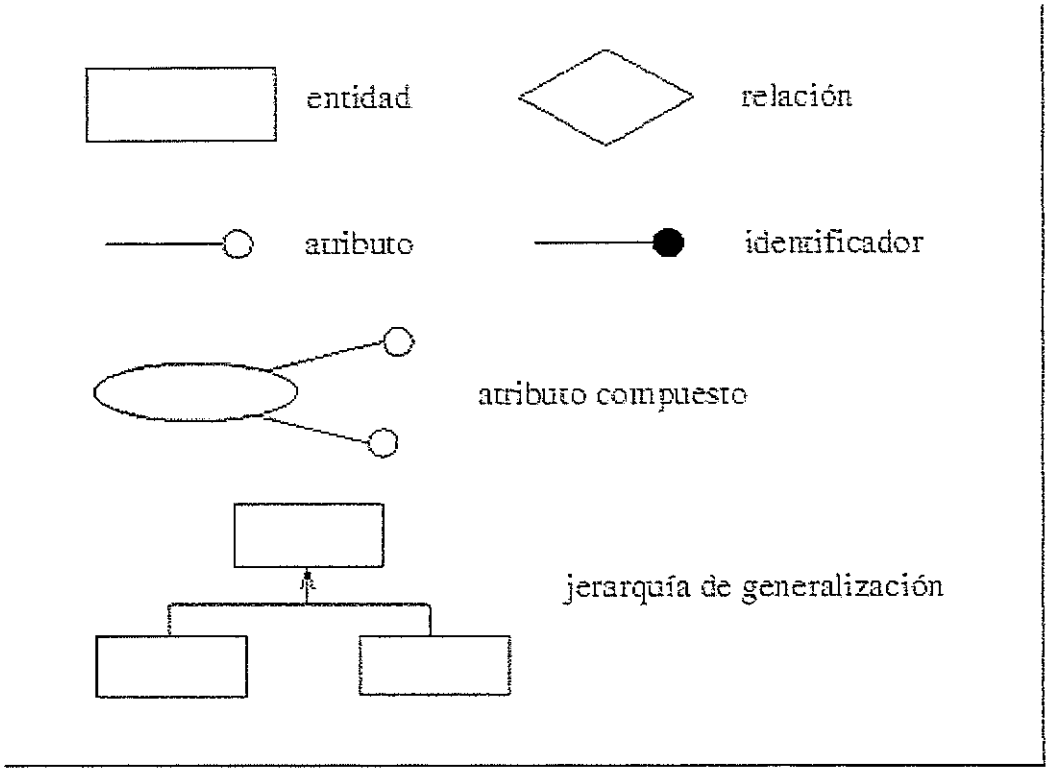
Antes de dar por finalizada la fase del diseño conceptual, se debe revisar el esquema conceptual local con el usuario. Este esquema está formado por el diagrama entidad-relación y toda la documentación que describe el esquema. Si se encuentra alguna anomalía, hay que corregirla haciendo los cambios oportunos, por lo que posiblemente haya que repetir alguno de los pasos anteriores. Este proceso debe repetirse hasta que se esté seguro de que el esquema conceptual es una fiel representación de la parte de la empresa que se está tratando de modelar.

2.2.1.1 El modelo Entidad – Relación

El modelo entidad-relación es el modelo conceptual más utilizado para el diseño conceptual de bases de datos. Fue introducido por Peter Chen en 1976. El modelo entidad-relación está formado por un conjunto de conceptos que permiten describir la realidad mediante un conjunto de representaciones gráficas y lingüísticas. Originalmente, el modelo entidad-relación sólo incluía los conceptos de entidad, relación y atributo. Más tarde, se añadieron otros conceptos, como los atributos compuestos y las

jerarquías de generalización, en lo que se ha denominado modelo entidad-*relación extendido.*

Gráfico 2



Conceptos del modelo entidad – relación extendido

Entidad

Cualquier tipo de objeto o concepto sobre el que se recoge información: cosa, persona, concepto abstracto o suceso. Por ejemplo: coches, casas, empleados, clientes, empresas, oficios, diseños de productos, conciertos, excursiones, etc. Las entidades se representan gráficamente mediante rectángulos y su nombre aparece en el interior. Un nombre de entidad sólo

puede aparecer una vez en el esquema conceptual. Hay dos tipos de entidades: fuertes y débiles. Una *entidad débil* es una entidad cuya existencia depende de la existencia de otra entidad. Una *entidad fuerte* es una entidad que no es débil.

Relación (interrelación)

Es una correspondencia o asociación entre dos o más entidades. Cada relación tiene un nombre que describe su función. Las relaciones se representan gráficamente mediante rombos y su nombre aparece en el interior. Las entidades que están involucradas en una determinada relación se denominan *entidades participantes*. El número de participantes en una relación es lo que se denomina *grado* de la relación. Por lo tanto, una relación en la que participan dos entidades es una relación *binaria*; si son tres las entidades participantes, la relación es *ternaria*; etc. Una *relación recursiva* es una relación donde la misma entidad participa más de una vez en la relación con distintos papeles. El nombre de estos papeles es importante para determinar la función de cada participación. La *cardinalidad* con la que una entidad participa en una relación especifica el número mínimo y el número máximo de correspondencias en las que puede tomar parte cada ocurrencia de dicha entidad. La participación de una entidad en una relación es *obligatoria (total)* si la existencia de cada una de sus ocurrencias requiere la existencia de, al menos, una ocurrencia de la otra entidad participante. Si no, la participación es *opcional (parcial)*. Las reglas

que definen la cardinalidad de las relaciones son las *reglas de negocio*. A veces, surgen problemas cuando se está diseñado un esquema conceptual. Estos problemas, denominados *trampas*, suelen producirse a causa de una mala interpretación en el significado de alguna relación, por lo que es importante comprobar que el esquema conceptual carece de dichas trampas. En general, para encontrar las trampas, hay que asegurarse de que se entiende completamente el significado de cada relación. Si no se entienden las relaciones, se puede crear un esquema que no represente fielmente la realidad. Una de las trampas que pueden encontrarse ocurre cuando el esquema representa una relación entre entidades, pero el camino entre algunas de sus ocurrencias es ambiguo. El modo de resolverla es reestructurando el esquema para representar la asociación entre las entidades correctamente. Otra de las trampas sucede cuando un esquema sugiere la existencia de una relación entre entidades, pero el camino entre una y otra no existe para algunas de sus ocurrencias. En este caso, se produce una pérdida de información que se puede subsanar introduciendo la relación que sugería el esquema y que no estaba representada.

Atributo

Es una característica de interés o un hecho sobre una entidad o sobre una relación. Los atributos representan las propiedades básicas de las entidades y de las relaciones. Toda la información extensiva es portada por los atributos. Gráficamente, se representan mediante bolitas que cuelgan de las

entidades o relaciones a las que pertenecen. Cada atributo tiene un conjunto de valores asociados denominado *dominio*. El dominio define todos los valores posibles que puede tomar un atributo. Puede haber varios atributos definidos sobre un mismo dominio. Los atributos pueden ser simples o compuestos. Un *atributo simple* es un atributo que tiene un solo componente, que no se puede dividir en partes más pequeñas que tengan un significado propio. Un *atributo compuesto* es un atributo con varios componentes, cada uno con un significado por sí mismo. Un grupo de atributos se representa mediante un atributo compuesto cuando tienen afinidad en cuanto a su significado, o en cuanto a su uso. Un atributo compuesto se representa gráficamente mediante un óvalo. Los atributos también pueden clasificarse en monovalentes o polivalentes. Un *atributo monovalente* es aquel que tiene un solo valor para cada ocurrencia de la entidad o relación a la que pertenece. Un *atributo polivalente* es aquel que tiene varios valores para cada ocurrencia de la entidad o relación a la que pertenece. A estos atributos también se les denomina *multivaluados*, y pueden tener un número máximo y un número mínimo de valores. La *cardinalidad* de un atributo indica el número mínimo y el número máximo de valores que puede tomar para cada ocurrencia de la entidad o relación a la que pertenece. Por último, los atributos pueden ser derivados. Un *atributo derivado* es aquel que representa un valor que se puede obtener a partir del valor de uno o varios atributos, que no necesariamente deben pertenecer a la misma entidad o relación.

Identificador

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

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

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

Jerarquía de generalización

Una entidad *E* es una generalización de un grupo de entidades E^1, E^2, ... E^n, si cada ocurrencia de cada una de esas entidades es también una ocurrencia de *E*. Todas las propiedades de la entidad genérica *E* son heredadas por las subentidades. Cada jerarquía es total o parcial, y exclusiva o superpuesta. Una jerarquía es *total* si cada ocurrencia de la entidad genérica corresponde al menos con una ocurrencia de alguna sub-entidad. Es *parcial* si existe alguna ocurrencia de la entidad genérica que no corresponde con ninguna ocurrencia de ninguna sub-entidad. Una jerarquía es *exclusiva* si cada ocurrencia de la entidad genérica corresponde, como

mucho, con una ocurrencia de una sola de las sub-entidades. Es *superpuesta* si existe alguna ocurrencia de la entidad genérica que corresponde a ocurrencias de dos o más subentidades diferentes. Un *subconjunto* es un caso particular de generalización con una sola entidad como sub-entidad. Un subconjunto siempre es una jerarquía parcial y exclusiva.

2.2.2 Metodología del diseño lógico

La metodología que se va a seguir para el diseño lógico en el modelo relacional consta de dos fases, cada una de ellas compuesta por varios pasos que se detallan a continuación.

2.2.2.1 Construir y validar los esquemas lógicos locales para cada vista de usuario.

- Convertir los esquemas conceptuales locales en esquemas lógicos locales.
- Derivar un conjunto de relaciones (tablas) para cada esquema lógico local.
- Validar cada esquema mediante la normalización.
- Validar cada esquema frente a las transacciones del usuario.
- Dibujar el diagrama entidad-relación.
- Definir las restricciones de integridad.
- Revisar cada esquema lógico local con el usuario correspondiente.

2.2.2.2 Construir y validar el esquema lógico global.

- Mezclar los esquemas lógicos locales en un esquema lógico global.
- Validar el esquema lógico global.
- Estudiar el crecimiento futuro.
- Dibujar el diagrama entidad-relación final.
- Revisar el esquema lógico global con los usuarios.

En la primera fase, se construyen los esquemas lógicos locales para cada vista de usuario y se validan. En esta fase se refinan los esquemas conceptuales creados durante el diseño conceptual, eliminando las estructuras de datos que no se pueden implementar de manera directa sobre el modelo que soporta el SGBD (sistema generador de base de datos), en el caso que nos ocupa, el modelo relacional. Una vez hecho esto, se obtiene un primer esquema lógico que se valida mediante la normalización y frente a las transacciones que el sistema debe llevar a cabo, tal y como se refleja en las especificaciones de requisitos de usuario. El esquema lógico ya validado se puede utilizar como base para el desarrollo de prototipos. Una vez finalizada esta fase, se dispone de un esquema lógico para cada vista de usuario que es correcto, comprensible y sin ambigüedad.

2.2.2.1.1 Convertir los esquemas conceptuales locales en esquemas lógicos locales

En este paso, se eliminan de cada esquema conceptual las estructuras de datos que los sistemas relacionales no modelan directamente:

(a)

Eliminar las relaciones de muchos a muchos, sustituyendo cada una de ellas por una nueva entidad intermedia y dos relaciones de uno a muchos de esta nueva entidad con las entidades originales. La nueva entidad será débil, ya que sus ocurrencias dependen de la existencia de ocurrencias en las entidades originales.

(b)

Eliminar las relaciones entre tres o más entidades, sustituyendo cada una de ellas por una nueva entidad (débil) intermedia que se relaciona con cada una de las entidades originales. La cardinalidad de estas nuevas relaciones binarias dependerá de su significado.

(c)

Eliminar las relaciones recursivas, sustituyendo cada una de ellas por una nueva entidad (débil) y dos relaciones binarias de esta nueva entidad con la entidad original. La cardinalidad de estas relaciones dependerá de su significado.

(d)

Eliminar las relaciones con atributos, sustituyendo cada una de ellas por una nueva entidad (débil) y las relaciones binarias correspondientes de esta nueva entidad con las entidades originales. La cardinalidad de estas relaciones dependerá del tipo de la relación original y de su significado.

(e)

Eliminar los atributos multievaluados, sustituyendo cada uno de ellos por una nueva entidad (débil) y una relación binaria de uno a muchos con la entidad original.

(f)

Revisar las relaciones de uno a uno, ya que es posible que se hayan identificado dos entidades que representen el mismo objeto (sinónimos). Si así fuera, ambas entidades deben integrarse en una sola.

(g)

Eliminar las relaciones redundantes. Una relación es redundante cuando se puede obtener la misma información que ella aporta

mediante otras relaciones. El hecho de que haya dos caminos diferentes entre dos entidades no implica que uno de los caminos corresponda a una relación redundante, eso dependerá del significado de cada relación.

Una vez finalizado este paso, es más correcto referirse a los esquemas conceptuales locales refinados como esquemas lógicos locales, ya que se adaptan al modelo de base de datos que soporta el SGBD escogido.

2.2.2.1.2 Derivar un conjunto de relaciones (tablas) para cada esquema lógico local

En este paso, se obtiene un conjunto de relaciones (tablas) para cada uno de los esquemas lógicos locales en donde se representen las entidades y relaciones entre entidades, que se describen en cada una de las vistas que los usuarios tienen de la empresa. Cada relación de la base de datos tendrá un nombre, y el nombre de sus atributos aparecerá, a continuación, entre paréntesis. El atributo o atributos que forman la clave primaria se subrayan. Las claves ajenas, mecanismo que se utiliza para representar las relaciones entre entidades en el modelo relacional, se especifican aparte indicando la relación (tabla) a la que hacen referencia. A continuación, se describe cómo las relaciones (tablas) del modelo relacional representan las entidades y relaciones que pueden aparecer en los esquemas lógicos.

(a)

Entidades fuertes. Crear una relación para cada entidad fuerte que incluya todos sus atributos simples. De los atributos compuestos incluir sólo sus componentes.

Cada uno de los identificadores de la entidad será una clave candidata. De entre las claves candidatas hay que escoger la clave primaria; el resto serán claves alternativas. Para escoger la clave primaria entre las claves candidatas se pueden seguir estas indicaciones:

- Escoger la clave candidata que tenga menos atributos.
- Escoger la clave candidata cuyos valores no tengan probabilidad de cambiar en el futuro.
- Escoger la clave candidata cuyos valores no tengan probabilidad de perder la unicidad en el futuro.
- Escoger la clave candidata con el mínimo número de caracteres (si es de tipo texto).
- Escoger la clave candidata más fácil de utilizar desde el punto de vista de los usuarios.

(b)

Entidades débiles. Crear una relación para cada entidad débil incluyendo todos sus atributos simples. De los atributos compuestos

incluir sólo sus componentes. Añadir una clave ajena a la entidad de la que depende. Para ello, se incluye la clave primaria de la relación que representa a la entidad padre en la nueva relación creada para la entidad débil. A continuación, determinar la clave primaria de la nueva relación.

(c)

Relaciones binarias de uno a uno. Para cada relación binaria se incluyen los atributos de la clave primaria de la entidad padre en la relación (tabla) que representa a la entidad hijo, para actuar como una clave ajena. La entidad hijo es la que participa de forma total (obligatoria) en la relación, mientras que la entidad padre es la que participa de forma parcial (opcional). Si las dos entidades participan de forma total o parcial en la relación, la elección de padre e hijo es arbitraria. Además, en caso de que ambas entidades participen de forma total en la relación, se tiene la opción de integrar las dos entidades en una sola relación (tabla). Esto se suele hacer si una de las entidades no participa en ninguna otra relación.

(d)

Relaciones binarias de uno a muchos. Como en las relaciones de uno a uno, se incluyen los atributos de la clave primaria de la entidad padre en la relación (tabla) que representa a la entidad hijo, para

actuar como una clave ajena. Pero ahora, la entidad padre es la de ``la parte del muchos" (cada padre tiene muchos hijos), mientras que la entidad hijo es la de la parte del uno (cada hijo tiene un solo padre).

(e)

Jerarquías de generalización. En las jerarquías, se denomina entidad padre a la entidad genérica y entidades hijo a las subentidades. Hay tres opciones distintas para representar las jerarquías. La elección de la más adecuada se hará en función de su tipo (total/parcial, exclusiva/superpuesta).

1. Crear una relación por cada entidad. Las relaciones de las entidades hijo heredan como clave primaria la de la entidad padre. Por lo tanto, la clave primaria de las entidades hijo es también una clave ajena al padre. Esta opción sirve para cualquier tipo de jerarquía, total o parcial y exclusiva o superpuesta.
2. Crear una relación por cada entidad hijo, heredando los atributos de la entidad padre. Esta opción sólo sirve para jerarquías totales y exclusivas.
3. Integrar todas las entidades en una relación, incluyendo en ella los atributos de la entidad padre, los atributos de todos los hijos y un atributo discriminativo para indicar el caso al cual

pertenece la entidad en consideración. Esta opción sirve para cualquier tipo de jerarquía. Si la jerarquía es superpuesta, el atributo discriminativo será multievaluado.

Una vez obtenidas las relaciones con sus atributos, claves primarias y claves ajenas, sólo queda actualizar el diccionario de datos con los nuevos atributos que se hayan identificado en este paso.

2.2.2.1.3 Validar cada esquema mediante la normalización

La normalización se utiliza para mejorar el esquema lógico, de modo que satisfaga ciertas restricciones que eviten la duplicidad de datos. La normalización garantiza que el esquema resultante se encuentra más próximo al modelo de la empresa, que es consistente y que tiene la mínima redundancia y la máxima estabilidad. La normalización es un proceso que permite decidir a qué entidad pertenece cada atributo. Uno de los conceptos básicos del modelo relacional es que los atributos se agrupan en relaciones (tablas) porque están relacionados a nivel lógico. En la mayoría de las ocasiones, una base de datos normalizada no proporciona la máxima eficiencia, sin embargo, el objetivo ahora es conseguir una base de datos normalizada por las siguientes razones:

- Un esquema normalizado organiza los datos de acuerdo a sus dependencias funcionales, es decir, de acuerdo a sus relaciones lógicas.

- El esquema lógico no tiene porqué ser el esquema final. Debe representar lo que el diseñador entiende sobre la naturaleza y el significado de los datos de la empresa. Si se establecen unos objetivos en cuanto a prestaciones, el diseño físico cambiará el esquema lógico de modo adecuado. Una posibilidad es que algunas relaciones normalizadas se desnormalicen. Pero la desnormalización no implica que se haya malgastado tiempo normalizando, ya que mediante este proceso el diseñador aprende más sobre el significado de los datos. De hecho, la normalización obliga a entender completamente cada uno de los atributos que se han de representar en la base de datos.
- Un esquema normalizado es robusto y carece de redundancias, por lo que está libre de ciertas anomalías que éstas pueden provocar cuando se actualiza la base de datos.
- Los equipos informáticos de hoy en día son mucho más potentes, por lo que en ocasiones es más razonable implementar bases de datos fáciles de manejar (las normalizadas), a costa de un tiempo adicional de proceso.
- La normalización produce bases de datos con esquemas flexibles que pueden extenderse con facilidad.

2.2.2.1.4 Validar cada esquema frente a las transacciones del usuario

El objetivo de este paso es validar cada esquema lógico local para garantizar que puede soportar las transacciones requeridas por los correspondientes usuarios. Estas transacciones se encontrarán en las especificaciones de requisitos de usuario. Lo que se debe hacer es tratar de realizar las transacciones de forma manual utilizando el diagrama entidad-relación, el diccionario de datos y las conexiones que establecen las claves ajenas de las relaciones (tablas). Si todas las transacciones se pueden realizar, el esquema queda validado. Pero si alguna transacción no se puede realizar, seguramente será porque alguna entidad, relación o atributo no se ha incluido en el esquema.

2.2.2.1.5 Dibujar el diagrama entidad-relación

En este momento, se puede dibujar el diagrama entidad-relación final para cada vista de usuario que recoja la representación lógica de los datos desde su punto de vista. Este diagrama habrá sido validado mediante la normalización y frente a las transacciones de los usuarios.

2.2.2.1.6 Definir las restricciones de integridad

Las restricciones de integridad son reglas que se quieren imponer para proteger la base de datos, de modo que no pueda llegar a un estado inconsistente. Hay cinco tipos de restricciones de integridad.

(a)

Datos requeridos. Algunos atributos deben contener valores en todo momento, es decir, no admiten nulos.

(b)

Restricciones de dominios. Todos los atributos tienen un dominio asociado, que es el conjunto de los valores que cada atributo puede tomar.

(c)

Integridad de entidades. El identificador de una entidad no puede ser nulo, por lo tanto, las claves primarias de las relaciones (tablas) no admiten nulos.

(d)

Integridad referencial. Es cuando una clave ajena enlaza cada tupla de la relación hijo con la tupla de la relación padre que tiene el mismo valor en su clave primaria. La integridad referencial dice que si una clave ajena tiene un valor (si es no nula), ese valor debe ser uno de los valores de la clave primaria a la que referencia. Hay varios

aspectos a tener en cuenta sobre las claves ajenas para lograr que se cumpla la integridad referencial.

1. ¿Admite nulos la clave ajena? Cada clave ajena expresa una relación. Si la participación de la entidad hijo en la relación es total, entonces la clave ajena no admite nulos; si es parcial, la clave ajena debe aceptar nulos.
2. ¿Qué hacer cuando se quiere borrar una ocurrencia de la entidad padre que tiene algún hijo? O lo que es lo mismo, ¿qué hacer cuando se quiere borrar una tupla que está siendo referenciada por otra tupla a través de una clave ajena? Hay varias respuestas posibles:
    - *Restringir:* no se pueden borrar tuplas que están siendo referenciadas por otras tuplas.
    - *Propagar:* se borra la tupla deseada y se propaga el borrado a todas las tuplas que le hacen referencia.
    - *Anular:* se borra la tupla deseada y todas las referencias que tenía se ponen, automáticamente, a nulo (esta respuesta sólo es válida si la clave ajena acepta nulos).
    - *Valor por defecto:* se borra la tupla deseada y todas las referencias toman, automáticamente, el valor por defecto (esta respuesta sólo es válida si se ha especificado un valor por defecto para la clave ajena).

- *No comprobar:* se borra la tupla deseada y no se hace nada para garantizar que se sigue cumpliendo la integridad referencial.

3. ¿Qué hacer cuando se quiere modificar la clave primaria de una tupla que está siendo referenciada por otra tupla a través de una clave ajena? Las respuestas posibles son las mismas que en el caso anterior. Cuando se escoge propagar, se actualiza la clave primaria en la tupla deseada y se propaga el cambio a los valores de clave ajena que le hacían referencia.

(e)

Reglas de negocio. Cualquier operación que se realice sobre los datos debe cumplir las restricciones que impone el funcionamiento de la empresa.

Todas las restricciones de integridad establecidas en este paso se deben reflejar en el diccionario de datos para que puedan ser tenidas en cuenta durante la fase del diseño físico.

2.2.2.1.7 Revisar cada esquema lógico local con el Usuario correspondiente

Para garantizar que cada esquema lógico local es una fiel representación de la vista del usuario lo que se debe hacer es comprobar con él que lo reflejado en el esquema y en la documentación es correcto y está completo.

2.2.2.1.8 Mezclar los esquemas lógicos locales en un Esquema lógico

En este paso, se deben integrar todos los esquemas locales en un solo esquema global. En un sistema pequeño, con dos o tres vistas de usuario y unas pocas entidades y relaciones, es relativamente sencillo comparar los esquemas locales, mezclarlos y resolver cualquier tipo de diferencia que pueda existir. Pero en los sistemas grandes, se debe seguir un proceso más sistemático para llevar a cabo este paso con éxito:

1. Revisar los nombres de las entidades y sus claves primarias.
2. Revisar los nombres de las relaciones.
3. Mezclar las entidades de las vistas locales.
4. Incluir (sin mezclar) las entidades que pertenecen a una sola vista de usuario.
5. Mezclar las relaciones de las vistas locales.
6. Incluir (sin mezclar) las relaciones que pertenecen a una sola vista de usuario.

7. Comprobar que no se ha omitido ninguna entidad ni relación.
8. Comprobar las claves ajenas.
9. Comprobar las restricciones de integridad.
10. Dibujar el esquema lógico global.
11. Actualizar la documentación.

2.2.2.1.9 Validar el esquema lógico global

Este proceso de validación se realiza, de nuevo, mediante la normalización y mediante la prueba frente a las transacciones de los usuarios. Pero ahora sólo hay que normalizar las relaciones que hayan cambiado al mezclar los esquemas lógicos locales y sólo hay que probar las transacciones que requieran acceso a áreas que hayan sufrido algún cambio.

2.2.2.1.10 Estudiar el crecimiento futuro

En este paso, se trata de comprobar que el esquema obtenido puede acomodar los futuros cambios en los requisitos con un impacto mínimo. Si el esquema lógico se puede extender fácilmente, cualquiera de los cambios previstos se podrá incorporar al mismo con un efecto mínimo sobre los usuarios existentes.

2.2.2.1.11 Dibujar el diagrama entidad-relación final

Una vez validado el esquema lógico global, ya se puede dibujar el diagrama entidad-relación que representa el modelo de los datos de la empresa que son de interés. La documentación que describe este modelo (incluyendo el esquema relacional y el diccionario de datos) se debe actualizar y completar.

2.2.2.1.12 Revisar el esquema lógico global con los usuarios

Una vez más, se debe revisar con los usuarios el esquema global y la documentación obtenida para asegurarse de que son una fiel representación de la empresa.

2.3 Lenguaje de Consultas

El lenguaje de consulta estructurado (SQL) es un lenguaje de base de datos normalizado, utilizado por los diferentes motores de bases de datos para realizar determinadas operaciones sobre los datos o sobre la estructura de los mismos. Pero como sucede con cualquier sistema de normalización hay excepciones para casi todo; de hecho, cada motor de bases de datos tiene sus peculiaridades y lo hace diferente de otro motor, por lo tanto, el lenguaje SQL normalizado (ANSI) no nos servirá para resolver todos los problemas, aunque si se puede asegurar que cualquier sentencia escrita en ANSI será interpretable por cualquier motor de datos.

2.3.1 Historia del Lenguaje de Consultas

La historia de SQL (que se pronuncia deletreando en inglés las letras que lo componen, es decir "ese-cu-ele" y no "siquel" como se oye a menudo) empieza en 1974 con la definición, por parte de Donald Chamberlin y de otras personas que trabajaban en los laboratorios de investigación de IBM, de un lenguaje para la especificación de las características de las bases de datos que adoptaban el modelo relacional. Este lenguaje se llamaba SEQUEL (Structured English Query Language) y se implementó en un prototipo llamado SEQUEL-XRM entre 1974 y 1975. Las experimentaciones con ese prototipo condujeron, entre 1976 y 1977, a una revisión del lenguaje (SEQUEL/2), que a partir de ese momento cambió de nombre por motivos legales, convirtiéndose en SQL. El prototipo (System R), basado en este lenguaje, se adoptó y utilizó internamente en IBM y lo adoptaron algunos de sus clientes elegidos. Gracias al éxito de este sistema, que no estaba todavía comercializado, también otras compañías empezaron a desarrollar sus productos relacionales basados en SQL. A partir de 1981, IBM comenzó a entregar sus productos relacionales y en 1983 empezó a vender DB2. En el curso de los años ochenta, numerosas compañías (por ejemplo Oracle y Sybase) comercializaron productos basados en SQL, que se convierte en el estándar industrial de hecho por lo que respecta a las bases de datos relacionales.

En 1986, el ANSI adoptó SQL (sustancialmente adoptó el dialecto SQL de IBM) como estándar para los lenguajes relacionales y en 1987 se transformó en estándar ISO. Esta versión del estándar va con el nombre de SQL/86. En los años siguientes, éste ha sufrido diversas revisiones que han conducido primero a la versión SQL/89 y, posteriormente, a la actual SQL/92.

El hecho de tener un estándar definido por un lenguaje para bases de datos relacionales abre potencialmente el camino a la intercomunicabilidad entre todos los productos que se basan en él. Desde el punto de vista práctico, por desgracia las cosas fueron de otro modo. Efectivamente, en general cada productor adopta e implementa en la propia base de datos sólo el corazón del lenguaje SQL (el así llamado Entry level o al máximo el Intermediate level), extendiéndolo de manera individual según la propia visión que cada cual tenga del mundo de las bases de datos.

Actualmente, está en marcha un proceso de revisión del lenguaje por parte de los comités ANSI e ISO, que debería terminar en la definición de lo que en este momento se conoce como SQL3. Las características principales de esta nueva encarnación de SQL deberían ser su transformación en un lenguaje stand-alone (mientras ahora se usa como lenguaje hospedado en otros lenguajes) y la introducción de nuevos tipos de datos más complejos que permitan, por ejemplo, el tratamiento de datos multimediales.

2.3.2 Componentes del SQL

El lenguaje SQL está compuesto por comandos, cláusulas, operadores y funciones de agregado. Estos elementos se combinan en las instrucciones para crear, actualizar y manipular las bases de datos.

2.3.2.1 Comandos

Existen dos tipos de comandos SQL:

- Los DLL que permiten crear y definir nuevas bases de datos, campos e índices.
- Los DML que permiten generar consultas para ordenar, filtrar y extraer datos de la base de datos.

Entre los comandos DLL tenemos a los siguientes:

CREATE: Utilizado para crear nuevas tablas, campos e índices.

DROP: Empleado para eliminar tablas e índices.

ALTER: Utilizado para modificar las tablas agregando campos o cambiando la definición de los campos.

A continuación se presentan los comandos DML utilizados para el manejo de consultas:

SELECT: Utilizado para consultar registros de la base de datos que satisfagan un criterio determinado.

INSERT: Utilizado para cargar lotes de datos en la base de datos en una única operación.

UPDATE: Utilizado para modificar los valores de los campos y registros especificados.

DELETE: Utilizado para eliminar registros de una tabla de una base de datos.

2.3.2.2 Cláusulas

Las cláusulas son condiciones de modificación utilizadas para definir los datos que desea seleccionar o manipular. Entre las cláusulas utilizadas para el manejo de consultas tenemos:

FROM: Utilizada para especificar la tabla de la cual se van a seleccionar los registros.

WHERE: Utilizada para especificar las condiciones que deben reunir los registros que se van a seleccionar.

GROUP BY: Utilizada para separar los registros seleccionados en grupos específicos.

HAVING: Utilizada para expresar la condición que debe satisfacer cada grupo.

ORDER BY: Utilizada para ordenar los registros seleccionados de acuerdo con un orden específico.

2.3.2.3 Operadores Lógicos

Los operadores lógicos son utilizados para enlazar condiciones que se expresan en la cláusula **WHERE** o para negar una condición específica. A continuación se presentan a los operadores lógicos para manejo de consultas:

AND: Es el "y" lógico. Evalúa dos condiciones y devuelve un valor de verdad sólo si ambas son ciertas.

OR: Es el "o" lógico. Evalúa dos condiciones y devuelve un valor de verdad si alguna de las dos es cierta.

NOT: Negación lógica. Devuelve el valor contrario de la expresión.

2.3.2.4 Operadores de Comparación

Estos operadores son utilizados para realizar comparaciones entre valores o variables en las condiciones que se expresan en la cláusula **WHERE**. A continuación se presentan los operadores de comparación utilizados en el manejo de consultas:

>	Mayor que
<	Menor que

>=	Mayor ó igual que
<=	Menor ó igual que
=	Igual que
<>	Distinto que
BETWEEN	Utilizado para especificar un rango de valores
LIKE	Utilizado en la comparación de un modelo
IN	Utilizado para especificar registros de una base de datos.

2.3.2.5 Funciones de Agregado

Las funciones de agregado se usan dentro de una cláusula SELECT en grupos de registros para devolver un único valor que se aplica a un grupo de registros. A continuación se menciona las funciones de agregado utilizadas para el manejo de consultas:

AVG: Utilizada para calcular el promedio de los valores de un campo determinado.

COUNT: Utilizada para devolver el número de registros de la selección.

SUM: Utilizada para devolver la suma de todos los valores de un campo determinado.

MAX: Utilizada para devolver el valor más alto de un campo especificado.

MIN: Utilizada para devolver el valor más bajo de un campo especificado.

2.3.3 Consultas de selección

Las consultas de selección se utilizan para indicar al motor de datos que devuelva información de las bases de datos, esta información es devuelta en forma de conjunto de registros que se pueden almacenar en un objeto recordset. Este conjunto de registros es modificable.

2.3.3.1 Estructura básica de las consultas

Para el manejo de consultas de bases de datos se debe seguir una estructura básica, que es la que especifica a continuación:

SELECT A1,A2,...,An

FROM r1,r2,...,rn

WHERE P

Donde Ai = atributo (Campo de la tabla)

ri = relación (Tabla)

P = predicado (condición)

Por ejemplo, para seleccionar todos los nombres de las personas que tengan el apellido CASTRO de la tabla persona se utiliza una consulta como la siguiente:

```
SELECT nombre

FROM persona

WHERE apellido = " CASTRO"
```

Es posible renombrar los atributos y las relaciones, a veces por conveniencia y otras veces por ser necesario, para esto usamos la cláusula **AS**. A continuación se presenta el ejemplo anterior con el uso de esta cláusula.

```
SELECT P.nombre AS [PRIMER NOMBRE]

FROM persona P

WHERE apellido = "CASTRO"
```

La complejidad de una consulta puede aumentar cada vez más. Esto depende de los requerimientos que se haga y del uso de los comandos y cláusulas que se necesiten. Por ejemplo, puede haber **subconsultas**, que no son nada más que una consulta dentro de otra; es decir, un SELECT dentro de otro. Puede utilizar tres formas de sintaxis para crear una subconsulta:

```
comparación [ANY | ALL | SOME] (instrucción sql)

expresión [NOT] IN (instrucción sql)

[NOT] EXISTS (instrucción sql)
```

En donde:

comparación

Es una expresión y un operador de comparación que compara la expresión con el resultado de la subconsulta.

expresión

Es una expresión por la que se busca el conjunto resultante de la subconsulta.

instrucción sql

Es una instrucción SELECT, que sigue el mismo formato y reglas que cualquier otra instrucción SELECT. Debe ir entre paréntesis.

Se puede utilizar una subconsulta en lugar de una expresión en la lista de campos de una instrucción SELECT o en una cláusula WHERE o HAVING. En una subconsulta, se utiliza una instrucción SELECT para proporcionar un conjunto de uno o más valores especificados para evaluar en la expresión de la cláusula WHERE o HAVING.

Se puede utilizar el predicado ANY o SOME, los cuales son sinónimos, para recuperar registros de la consulta principal, que satisfagan la comparación con cualquier otro registro recuperado en la subconsulta. El ejemplo siguiente devuelve todos los productos cuyo precio unitario es mayor que el

de cualquier producto vendido con un descuento igual o mayor al 25 por ciento:

```
SELECT * FROM Productos WHERE PrecioUnidad > ANY
(SELECT PrecioUnidad FROM DetallePedido WHERE Descuento >= 0 .25);
```

El predicado ALL se utiliza para recuperar únicamente aquellos registros de la consulta principal que satisfacen la comparación con todos los registros recuperados en la subconsulta. Si se cambia ANY por ALL en el ejemplo anterior, la consulta devolverá únicamente aquellos productos cuyo precio unitario sea mayor que el de todos los productos vendidos con un descuento igual o mayor al 25 por ciento. Esto es mucho más restrictivo.

El predicado IN se emplea para recuperar únicamente aquellos registros de la consulta principal para los que algunos registros de la subconsulta contienen un valor igual. El ejemplo siguiente devuelve todos los productos vendidos con un descuento igual o mayor al 25 por ciento:

```
SELECT * FROM Productos WHERE IDProducto IN
(SELECT IDProducto FROM DetallePedido WHERE Descuento >=  0.25);
```

Inversamente se puede utilizar NOT IN para recuperar únicamente aquellos registros de la consulta principal para los que no hay ningún registro de la subconsulta que contenga un valor igual.

El predicado EXISTS (con la palabra reservada NOT opcional) se utiliza en comparaciones de verdad/falso para determinar si la subconsulta devuelve algún registro. Supongamos que deseamos recuperar todos aquellos clientes que hayan realizado al menos un pedido:

SELECT Clientes.Compañía, Clientes.Teléfono FROM Clientes WHERE EXISTS

(SELECT FROM Pedidos WHERE Pedidos.IdPedido = Clientes.IdCliente)

Esta consulta es equivalente a esta otra:

SELECT Clientes.Compañía, Clientes.Teléfono FROM Clientes WHERE IdClientes IN

(SELECT Pedidos.IdCliente FROM Pedidos)

Se puede utilizar también alias del nombre de la tabla en una subconsulta para referirse a tablas listadas en la cláusula FROM fuera de la subconsulta. El ejemplo siguiente devuelve los nombres de los empleados cuyo salario es

igual o mayor que el salario medio de todos los empleados con el mismo título. A la tabla Empleados se le ha dado el alias T1:

SELECT Apellido, Nombre, Titulo, Salario FROM Empleados AS T1

WHERE Salario >= (SELECT Avg(Salario) FROM Empleados

WHERE T1.Titulo = Empleados.Titulo) ORDER BY Titulo;

2.3.3.2 Consultas de combinación entre tablas

Las vinculaciones entre tablas se realizan mediante la cláusula INNER que combina registros de dos tablas siempre que haya concordancia de valores en un campo común. Su sintaxis es la siguiente;

SELECT campos FROM tb1 INNER JOIN tb2 ON tb1.campo1 comp tb2.campo2

En donde:

tb1, tb2: Son los nombres de las tablas desde las que se combinan los registros.

campo1, campo2: Son los nombres de los campos que se combinan. Si no son numéricos, los campos deben ser del mismo tipo de datos y contener el mismo tipo de datos, pero no tienen que tener el mismo nombre.

Comp: Es cualquier operador de comparación relacional: =, <, >, <=, >=, o <>.

Se puede utilizar una operación INNER JOIN en cualquier cláusula FROM. Esto crea una combinación por equivalencia, conocida también como unión interna. Las combinaciones Equi son las más comunes; éstas combinan los registros de dos tablas siempre que haya concordancia de valores en un campo común a ambas tablas. Se puede utilizar INNER JOIN con las tablas Departamentos y Empleados para seleccionar todos los empleados de cada departamento. Por el contrario, para seleccionar todos los departamentos (incluso si alguno de ellos no tiene ningún empleado asignado) se emplea LEFT JOIN o todos los empleados (incluso si alguno no está asignado a ningún departamento), en este caso RIGHT JOIN.

Si se intenta combinar campos que contengan datos Memo u Objeto OLE, se produce un error. Se pueden combinar dos campos numéricos cualesquiera, incluso si son de diferente tipo de datos. Por ejemplo, puede combinar un campo Numérico para el que la propiedad Size de su objeto Field está establecida como Entero, y un campo Contador. El ejemplo siguiente muestra cómo podría combinar las tablas Categorías y Productos basándose en el campo IDCategoria:

```
SELECT Nombre_Categoría, NombreProducto
FROM Categorias INNER JOIN Productos
ON Categorias.IDCategoria = Productos.IDCategoria;
```

2.3.3.3 Consultas de autocombinación

La autocombinación se utiliza para unir una tabla consigo misma, comparando valores de dos columnas con el mismo tipo de datos. La sintaxis es la siguiente:

SELECT alias1.columna, alias2.columna, ...

FROM tabla1 as alias1, tabla2 as alias2

WHERE alias1.columna = alias2.columna

AND otras condiciones

Por ejemplo, para visualizar el número, nombre y puesto de cada empleado, junto con el número, nombre y puesto del supervisor de cada uno de ellos se utilizaría la siguiente sentencia:

SELECT t.num_emp, t.nombre, t.puesto, t.num_sup,s.nombre, s.puesto

FROM empleados AS t, empleados AS s WHERE t.num_sup = s.num_emp

2.3.3.4 Consultas de combinaciones no comunes

La mayoría de las combinaciones están basadas en la igualdad de valores de las columnas que son el criterio de la combinación. Las no comunes se basan en otros operadores de combinación, tales como NOT, BETWEEN, <>, etc.

Por ejemplo, para listar el grado salarial, nombre, salario y puesto de cada empleado ordenando el resultado por grado y salario habría que ejecutar la siguiente sentencia:

```
SELECT grados.grado,empleados.nombre, empleados.salario,
empleados.puesto
FROM empleados, grados
WHERE empleados.salario
BETWEEN grados.salarioinferior AND grados.salariosuperior
ORDER BY grados.grado, empleados.salario
```

CAPITULO 3

3.1 Administración del tutorial

En este capítulo se hará referencia a la explicación detallada de cada uno de los componentes del tutorial, comenzando por los objetivos que persigue, el alcance del mismo, tablas y campos respectivos utilizados, diagrama entidad – relación de la base de datos planteada, diseño de la interfaz del usuario y lenguajes de programación utilizados para el desarrollo del tutorial.

3.1.1 Definición y objetivos del tutorial

Las Tecnologías de Información se han desarrollado en un corto período de tiempo llegando a ser una de las fuerzas conductoras de la economía mundial. Los países que han sido beneficiados por el potencial de las modernas tecnologías de Información prometen superar los tradicionales obstáculos, propios de las infraestructuras insuficientes, para convertir a éstas en una eficiente vía de apoyo a los principales objetivos que tiene la Cooperación al Desarrollo Internacional, los mismos que son: reducción de la pobreza, salud y educación.

En la actualidad hay diversidad de medios para la enseñanza. Uno de esos medios son los sistemas multimedia. Al pasar los años los sistemas

multimedia han ido captando la atención cada vez más de los educadores y estudiantes, por la facilidad y diversidad de formas de acceder a mucha información. Hoy en día los sistemas multimedia no sólo son simples gestores de grandes cantidades de información con una interfaz más amigable, sino que se presentan también como un medio con el cual los usuarios pueden interactuar de diversas formas permitiendo así al sistema ser una especie de tutor o profesor que le enseña a los usuarios a comportarse o manejar ciertas circunstancias reales.

Hoy en día los tutoriales representan una gran herramienta para el aprendizaje de cualquier tipo. Es así que existe muchos tutoriales que enseñan a los usuarios a manejar negocios, programar en diversos lenguajes, diseñar páginas Web, aprender matemáticas, aprender a cocinar, etc. A nivel educativo resulta muy interesante contar con el aporte de tutoriales para que los alumnos tengan otro recurso más para acceder y obtengan mayores conocimientos que los impartidos en un aula de clases.

En el ámbito empresarial el manejo de bases de datos resulta imprescindible porque se maneja grandes volúmenes de información que si no se los manejara con un sistema de bases de datos resultaría dificultoso su tratamiento y proceso. Es por eso que se debe tener gran énfasis en la

enseñanza de diseñar e implementar bases de datos para luego poder ejecutar las consultas necesarias.

El diseño del tutorial que se propone en esta tesis le permite a los usuarios evaluar sus conocimientos en el manejo de consultas de bases de datos, permitiendo con esto ganar en experiencia en el manejo de consultas. Se asegura que las personas que hagan uso de este tutorial van a sentirse incentivadas al aprendizaje del manejo de consultas de bases de datos. Además este tutorial presenta información adicional para el manejo de consultas que puede resultar interesante e importante para los usuarios.

3.1.2 Producto

- **Nombre:** Tutorial interactivo de consultas de SQL Server
- **Slogan:** TICSS (Tutorial interactivo de consultas de bases de datos SQL Server).

3.1.3. Misión

Permitir a cualquier usuario autoevaluar sus conocimientos en el manejo de consultas de bases de datos, obteniendo una calificación de acuerdo a su esfuerzo y la frecuencia con la que ha hecho uso del evaluador que presenta el tutorial, además de presentar información acerca de las sintaxis y ejemplos de las instrucciones básicas necesarias para obtener una consulta correcta.

3.1.4. Visión

- Incentivar el aprendizaje del manejo de consultas de bases de datos a través del evaluador, ya que es una herramienta muy importante con la que las personas pueden competir y ganarse un puesto en este campo.
- Permitir a los usuarios aprender las sintaxis de las instrucciones básicas para el manejo de consultas.
- Permitir a los usuarios autoevaluarse, obteniendo una nota por su conocimiento y esfuerzo; como si fuera una prueba tomada por escrito.

3.1.5 Alcance

- El alcance que el tutorial llegará a cubrir es:
- EL tutorial tendrá una interfaz amigable para que los usuarios se sientan incentivados a autoevaluar sus conocimientos en el manejo de consultas.
- Emitir gráficos estadísticos acerca de los resultados de la evaluación de cada usuario.
- Emitir gráficos estadísticos acerca de los visitantes del tutorial que hagan uso el evaluador.

- El sitio Web señalará que los derechos intelectuales de este tutorial corresponden exclusivamente a la Escuela Superior Politécnica del Litoral.

3.1.6 En la actualidad

Hoy en día podemos ver que en el país se vuelve más común el desarrollo de tutoriales para el aprendizaje de diversas asignaturas, tales como tutoriales de matemáticas, física, química, gramática, ortografía, etc. Es muy importante que las autoridades educativas pongan énfasis en el desarrollo y uso de los mismos, ya que representan un complemento al trabajo que desempeñan los educadores desde un aula de clases. Muchas veces ha habido personas que simplemente aprenden algo con la ayuda de un tutorial, sin necesidad de un profesor, por lo que resulta muy interesante el desarrollo de cada vez más tutoriales que ayuden a mejorar el nivel de educación en nuestro país.

3.1.7. Ventajas y desventajas

- **Ventajas**
- El evaluador interactivo que presenta el tutorial es la ventaja más importante a señalar, ya que permite evaluar y calificar el conocimiento en el lenguaje de consultas del usuario.

- Resultados de las evaluaciones de consultas a través de gráficos estadísticos que resaltan el conocimiento o desconocimiento del lenguaje de consultas.
- Presentación de sintaxis y ejemplos de cada una de las instrucciones básicas para el manejo de consultas.
- Reconocimiento y conocimiento de la importancia y utilidad de un Tutorial como este, tanto en nuestro país como en el exterior.
- Incentivar a realizar este tipo de tutoriales que ayuden al desarrollo no sólo del manejo de consultas de bases de datos sino al aprendizaje de otras herramientas informáticas muy importantes.
- Presentación de interfaces muy amigables para el usuario que pueden resultar de mejor agrado que un libro de consultas o un simple tutorial de presentación de datos.

Desventajas.

- El segmento del mercado que tiene conocimientos y puede acceder a Internet es reducido en nuestro país.
- La infocultura que tienen las personas es pobre con respecto a la importancia de un tutorial en el momento del aprendizaje.
- Falta de incentivo de educadores para el desarrollo de este tipo de trabajos.

3.1.8. Análisis F.O.D.A.

Análisis que determina las fortalezas y debilidades del tutorial en el ambiente externo y las oportunidades y amenazas en el ambiente interno.

Fortalezas

- El aprendizaje a través de tutoriales puede ser un complemento del que se puede impartir en las aulas de clases.
- La comunidad está percibiendo los beneficios que ofrece el Internet tal como lo son los tutoriales y la infinidad de cosas que se pueden encontrar en ella.

Debilidades

- Desconocimiento de la existencia de este tipo de sistemas en el sector de la educación.
- Incentivo a los alumnos para que se conviertan en continuos usuarios para el aprendizaje del manejo de consultas.

Oportunidades

- Gran número de visitantes en la Web.
- Dinamismo entre usuario y tutorial.
- Posibilidad de crecer como profesionales dando a conocer nuestro trabajo.

Amenazas

- Constante actualización del tutorial con nuevas bases de datos que sirvan de prueba para la generación de consultas que puedan ser evaluadas.
- Falta de reconocimiento en el ámbito educativo.

3.2 Diseño del Sistema

En esta parte analizaremos el diseño tanto de la base de datos utilizada, con sus respectivas tablas y campos, relaciones y otros; así como también las interfaces de usuarios creadas para la presentación del tutorial.

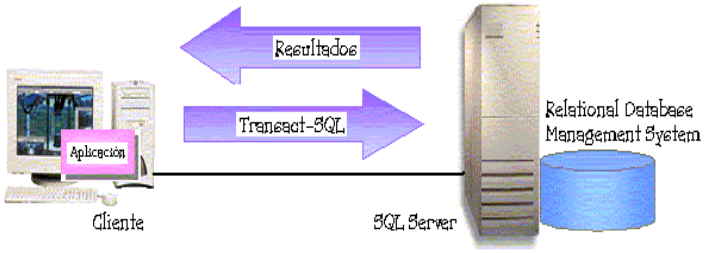
3.2.1 Diseño de la base de datos en SQL Server

El motor de la Base de Datos que se seleccionó fue SQL Server por su facilidad de uso en Intranet/Internet y por ser una base de datos de capacidad media, entre otros atributos que con el desarrollo de éste capítulo se irán detallando.

SQL Server es un sistema administrador para Bases de Datos relacionales basadas en la arquitectura Cliente / Servidor (RDBMS) que usa Transact-SQL para mandar peticiones entre un cliente y el SQL Server. SQL Server usa la arquitectura Cliente / Servidor para separar la carga de trabajo en

tareas que corran en computadoras tipo Servidor y tareas que corran en computadoras tipo Cliente:

Gráfico 3



Elaborado por Ing. Rubén Álvarez

- El Cliente es responsable de la parte lógica y de presentar la información al usuario. Generalmente, el cliente corre en una o más computadoras Cliente, aunque también puede correr en una computadora Servidor con SQL Server.

- SQL Server administra Bases de Datos y distribuye los recursos disponibles del servidor (tales como memoria, operaciones de disco, etc.) entre las múltiples peticiones.

- La arquitectura Cliente /Servidor nos permite desarrollar aplicaciones para realizar en una variedad de ambientes.

Una de los beneficios de utilizar como motor de base de datos a SQL SERVER es el (RDBMS) ya que es el responsable de:

- Mantener las relaciones entre la información y la Base de Datos.
- Asegurarse de que la información es almacenada correctamente, es decir, que las reglas que definen las relaciones ente los datos no sean violadas.
- Recuperar toda la información en un punto conocido en caso de que el sistema falle.

3.2.2 Definición de las tablas y campos que conforman la base de datos

Las tablas con sus respectivos campos que conforman nuestra Base de Datos son:

Tabla 1
Definición de la tabla "base"

Tabla: Base				
Descripción: Indica los nombres de las bases de datos de prueba planteadas		**Autor de Creación:** David Rugel González		**Tabla:** 1/1
Nombre del campo	**Tipo de datos**	**Long**	**Descripción**	**Null**
Idcodigo	int	4	Código de la base de prueba	Not null
Descripcion	nvarchar	20	Nombre de la base de prueba	Not null

Tabla 2
Definición de la tabla "tabla"

Tabla: Tabla				
Descripción: Indica las tablas de cada una de las bases de datos de prueba		**Autor de Creación:** David Rugel González		**Tabla:** 1/2
Nombre del campo	**Tipo de datos**	**Long**	**Descripción**	**Null**
Idtabla	int	4	Código de la tabla	Not null
Nombre	nvarchar	20	Nombre de la tabla	Not null
Descripción	nvarchar	30	Descripción detallada del nombre de la tabla	Not null
Base_id	Int	4	Código de la base a la que pertenece esta tabla	Not null

Tabla 3
Definición de la tabla "columnas"

Tabla: Columnas				
Descripción: Indica la descripción de las columnas de cada tabla de todas las bases de prueba		**Autor de Creación:** David Rugel González		**Tabla:** 1/3
Nombre del campo	**Tipo de datos**	**Long**	**Descripción**	**Null**
Id_columna	int	4	Código de la columna	Not null
Tabla:id	Int	4	Código de la tabla a la que pertenece la columna	Not null
Nombre	nvarchar	50	Nombre de la columna	Not null
Descripción	nvarchar	50	Descripción detallada de la	Not null

			columna	
T_dato	Int	4	Código del tipo de dato al que pertenece la columna	Not null

Tabla 4
Definición de la tabla "tipo_dato"

Tabla: Tipo_dato				
Descripción: Indica los diferentes tipos de datos existentes		**Autor de Creación:** David Rugel González		**Tabla:** 1/4
Nombre del campo	**Tipo de datos**	**Long**	**Descripción**	**Null**
Id_tipo	int	4	Código del tipo de datos	Not null
Descripcion	nvarchar	20	Descripción detallada del tipo de dato	Not null

Tabla 5
Definición de la tabla "Valores"

Tabla: Valores				
Descripción: Indica valores de prueba para cada columna		**Autor de Creación:** David Rugel González		**Tabla:** 1/5
Nombre del campo	**Tipo de datos**	**Long**	**Descripción**	**Null**
Idvalor	int	4	Código del valor	Not null
Columna_id	Int	4	Código de la columna a la q pertenece el valor	Not null
valor	nvarchar	25	Valor de prueba que puede ser texto o números	Null

Tabla 6
Definición de la tabla "Temporal"

Tabla: Temporal				
Descripción: Indica todos los códigos de los datos necesarios para la consulta		**Autor de Creación:** David Rugel González		**Tabla:** 1/6
Nombre del campo	**Tipo de datos**	**Long**	**Descripción**	**Null**
Base	Int	4	Código de la base para consulta	Null
Tabla	int	4	Código de la tabla para consulta	Null
Columna	int	4	Código de la columna para la consulta	Null
Valor	int	4	Código del valor específico para la consulta	Null
Njoin	int	4	Código de la tabla con la que se relaciona	Null
Hijo	int	4	Código de la tabla hijo. Si no hay hijo el valor es 0	Null

Tabla 7
Definición de la tabla "Temporal_d"

Tabla: Temporal_d				
Descripción: Indica todas las descripciones de los datos necesarios para la consulta		**Autor de Creación:** David Rugel González		**Tabla:** 1/7
Nombre del campo	**Tipo de datos**	**Long**	**Descripción**	**Null**

Based	nvarchar	50	Descripción de la base para consulta	Not Null
ntablad	nvarchar	50	Descripción de la tabla para consulta	Not Null
Columnad	nvarchar	50	Descripción de la columna para la consulta	Not Null
Valord	nvarchar	50	Descripción del valor específico para la consulta	Not Null
joind	nvarchar	50	Descripción de la tabla con la que se relaciona	Not Null
Hijo	nvarchar	50	Descripción de la tabla hijo. Si no hay hijo el valor es 0	Null

Tabla 8
Definición de la tabla "Tipo_error"

Tabla: Tipo_error				
Descripción: Indica los tipos de error existentes para la evaluación de la consulta			**Autor de Creación:** David Rugel González	**Tabla:** 1/8
Nombre del campo	**Tipo de datos**	**Long**	**Descripción**	**Null**
Errorid	int	4	Código del error	Not null
Tipo_er	nvarchar	50	Descripción del tipo de error	Not null

Tabla 9
Definición de la tabla "C_errores"

Tabla: C_errores				
Descripción: Indica la cantidad de errores de acuerdo al tipo		**Autor de Creación:** David Rugel González		**Tabla:** 1/9
Nombre del campo	**Tipo de datos**	**Long**	**Descripción**	**Null**
Tipoid	int	4	Código del tipo de error relacionado	Not null
Cantidad_error	Int	4	Cantidad de errores de acuerdo al tipo especificado	Not null

Tabla 10
Definición de la tabla "Refer_f_k"

Tabla: Refer_f_k				
Descripción: Indica las tablas que están relacionadas (que tienen join con otra tabla)		**Autor de Creación:** David Rugel González		**Tabla:** 1/10
Nombre del campo	**Tipo de datos**	**Long**	**Descripción**	**Null**
Tabla_hijo_id	int	4	Código de la tabla reconocida como padre	Null
Tabla_padre_id	Int	4	Código de la tabla reconocida como hijo	Null

Tabla 11
Definición de la tabla "Cabecera_visita"

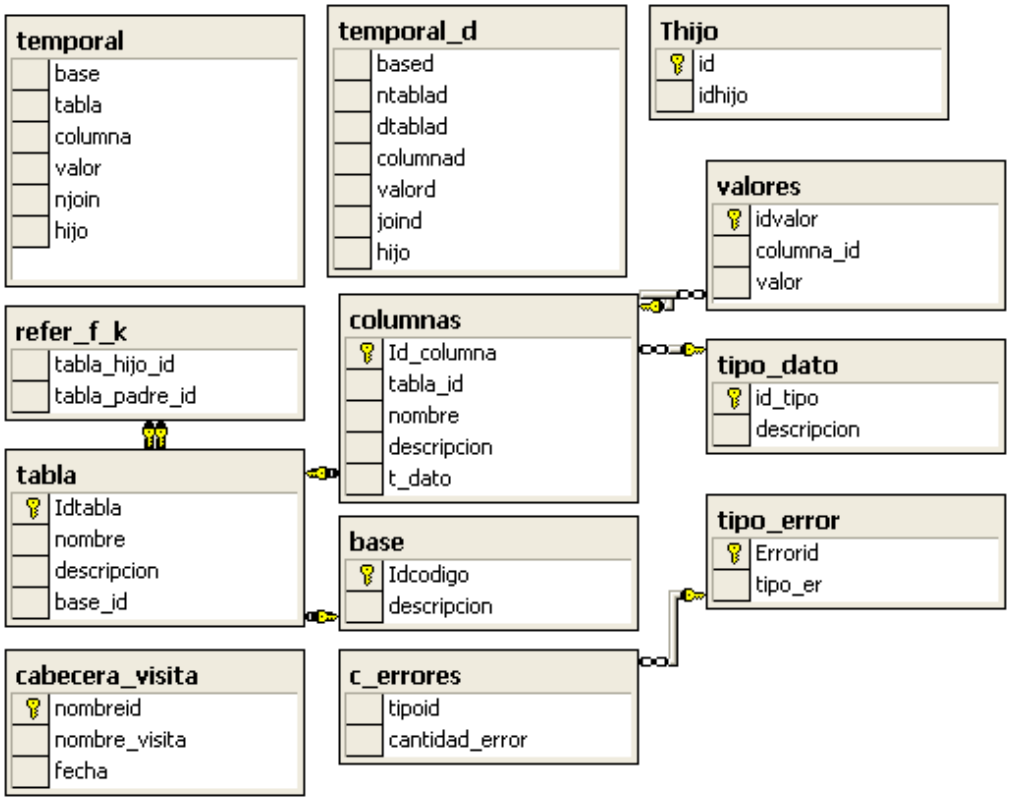
Tabla: Cabecera_visita				
Descripción: Indica datos de la persona que utiliza el evaluador		**Autor de Creación:** David Rugel González		**Tabla:** 1/11
Nombre del campo	**Tipo de datos**	**Long**	**Descripción**	**Null**
Nombreid	int	4	Código del tipo de error relacionado	Not null
Nombre_visita	nvarchar	4	Cantidad de errores de acuerdo al tipo especificado	Not null
Fecha	Datetime			Null

Tabla 12
Definición de la tabla "Thijo"

Tabla: Thijo				
Descripción: Indica el código de la tabla que trabaja como hijo de otra tabla		**Autor de Creación:** David Rugel González		**Tabla:** 1/12
Nombre del campo	**Tipo de datos**	**Long**	**Descripción**	**Null**
Id	numeric	18, 0	Código generado para el hijo	Not null
Idhijo	numeric	18, 0	Código de la tabla que actúa como hijo	Not null

3.2.3 Diagrama entidad – relación del tutorial

Gráfico 4
Diagrama entidad – relación de la base de datos del sistema



3.2.4 Esquema de la aplicación WEB

La aplicación Web no se trata simplemente de páginas con imágenes y descripciones. Cuando hablamos de construir aplicaciones, a lo que nos referimos es a la construcción de sitios que hagan algo; que nos permitan introducir información y respondan inteligentemente a las solicitudes.

Entre los servidores Web más utilizados se encuentra Microsoft Internet Information Server (IIS), que es el que utilizamos para la elaboración del

tutorial. Un servidor de aplicaciones es un software que ayuda al servidor Web a procesar las páginas que contienen scripts o etiquetas del lado del servidor. Cuando se solicita al servidor una página de este tipo, el servidor Web pasa la página al servidor de aplicaciones para su procesamiento antes de enviarla al navegador. Esta no se puede comunicar directamente con una base de datos porque el formato propietario de esta última impide que se descifren los datos, de una forma bastante similar a cuando la información de un documento de Microsoft Word abierto en el Bloc de Notas o BBEdit queda ininteligible. El servidor de aplicaciones sólo se puede comunicar con la base de datos a través de un controlador que actúe de intermediario con la base de datos: el software actúa entonces como un intérprete entre el servidor de aplicaciones y la base de datos.

Para comunicarse con la base de datos del tutorial, la aplicación Web utiliza un controlador ODBC y proveedores de OLE DB para SQL Server.
Una vez que el controlador establece la comunicación, la consulta se ejecuta en la base de datos y se crea un juego de registros:

Una ***consulta de base de datos*** es la operación mediante la cual se extrae un juego de registros de una base de datos.
Un ***juego de registros*** es un conjunto de datos extraídos de una o varias tablas de una base de datos.

Se puede utilizar prácticamente cualquier base de datos con una aplicación Web, siempre y cuando se haya instalado el controlador correcto de base de datos en el servidor.

Para el desarrollo del tutorial interactivo se utilizó una base de datos basada en servidor, como las que permite crear Microsoft SQL Server.

3.2.5 Diseño de la interfaz del usuario

Para el diseño de las interfaces para el usuario consideramos el uso de Dreamweaver MX por su facilidad de uso para generar aplicaciones web dinámicas con ASP, HTML y demás tecnologías para internet. Cabe indicar que DREAMWEAVER MX generó casi el 80% del código de la aplicación, el 20% restante lo conforman las consultas y procedimientos que se necesitaron para la generación y evaluación de las consultas.

3.3 Implementación del sistema

La implementación del sistema abarca desde la instalación, configuración, carga de datos, capacitación mediante manual de instalación, pruebas que fueron permitidas gracias a la utilidad del (IIS).

3.3.1 Software utilizado en la implementación del sistema

- Para la implementación de la base de datos del sistema se utilizó el Gestor de Base de Datos MSSQL Server 2000.

- En el diseño de la parte visual de las páginas Web, se utilizó Macromedia Flash MX y Macromedia Fireworks MX.
- Para el desarrollo de las páginas .htm y .asp, se utilizó el editor de páginas web Macromedia Dreamweaver MX.

3.4 Evaluación del sistema

Partiendo del hecho de que la educación es uno de los aspectos más importantes para el desarrollo de un país, un tutorial representa un gran aporte en el ámbito educacional, ya que contribuye con conocimientos para el aprendizaje de diversas asignaturas en diferentes áreas.

Se mantiene como objetivo principal el incentivar al aprendizaje de consultas de bases de datos, ya que en el ámbito profesional representa una herramienta muy poderosa para el tratamiento y proceso de información necesaria para poder tomar decisiones.

Se trató de que el tutorial abarque el uso de las instrucciones más básicas para que no resulte tan complicado para el usuario la sentencia que tenga que ejecutar para que luego sea evaluada. Por eso se puede considerar a este tutorial como un aporte valioso a la educación.

CONCLUSIONES

Después de la investigación realizada, el diseño e implementación del TICSS podemos hacer las siguientes conclusiones:

1. Las tecnologías de información han avanzado en gran magnitud al pasar los años.

2. Los sistemas multimedia han incursionado cada vez más fuerte en diversos ámbitos, a nivel educativo, a nivel empresarial, entre otros.

3. Los tutoriales independientemente del tipo que sean, representan un aporte muy importante a la educación.

4. Los espacios que puede abarcar un tutorial son ilimitados, por lo que resulta sencillo para cualquier persona, hacer uso de los mismos.

5. La mayoría de los tutoriales que existen en la Web son de tipo informativo, por lo que sólo se remiten a presentar información.

6. Existen pocos tutoriales que le permiten a los usuarios interactuar directamente, simulando un proceso Profesor – Alumno.

7. Gran parte de los estudiantes universitarios hace uso de tutoriales que les permitan complementar un estudio o darle solución a problemas planteados.

8. En colegios, los educadores hacen uso de ciertos tutoriales para complementar el aprendizaje de los alumnos que es impartido en un aula de clases.

9. El TICSS planteado en este trabajo cumplió con todos los requerimientos que se propusieron al inicio de la investigación.

10. El evaluador dinámico de consultas planteado en el TICSS, funciona de acuerdo a lo que se propuso; es decir hace la función de un compilador.

11. Los resultados que se presentan luego del proceso de evaluación de consultas son absolutamente confiables.

12. Los gráficos estadísticos presentados por el evaluador, representan la realidad de los conocimientos del usuario en el manejo de consultas.

RECOMENDACIONES

Al concluir este trabajo y analizar cada uno de los pasos que se siguió y cada uno de los puntos que se desarrolló se pueden realizar las siguientes recomendaciones:

1. Que las autoridades educativas incentiven al diseño e implementación de tutoriales que aporten al desarrollo de la educación en nuestro país.

2. Que los educadores, tanto de colegios y universidades enseñen a los alumnos que los tutoriales representan una fuente importante para complementar su estudio.

3. Que se considere el desarrollo de tutoriales interactivos como el TICSS para futuros trabajos a desempeñarse, considerando la interacción existente entre usuario y sistema.

4. Que se considere por parte de las instituciones educativas, que el tratamiento y proceso de información es muy importante en todo ámbito para que incentive al aprendizaje del diseño de bases de datos y manejo de consultas.

5. Que se proponga el desarrollo de tutoriales interactivos como tesis de estudiantes, cada vez con más complejidad que ayude al desarrollo intelectual de quienes hagan uso del mismo.

6. Que las instituciones privadas financien este tipo de proyectos, para el desarrollo intelectual de estudiantes que luego formarán parte del ámbito de la vida profesional.

7. Que el Gobierno Nacional haga un reconocimiento a este tipo de proyectos que permiten el desarrollo de la educación y formar profesionales más preparados y con mejores bases.

ANEXOS

ANEXO 1

INSTALACIÓN DE LA BASE DE DATOS EN SQL SERVER 2000

Notas de instalación de una intranet

Existen dos formas de instalar la base de datos con SQL Server 2000 con el ***Backup de la base de datos*** o con el ***Script SQL de la base de datos***.

La instalación de la base de datos y del sistema sobre Internet es específica de cada Hosting.

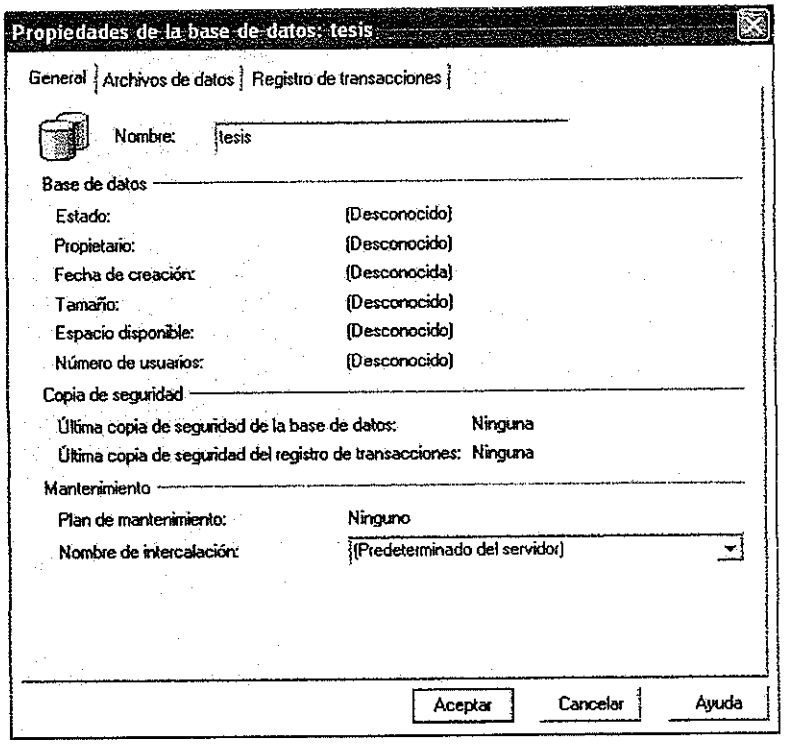
Backup de la base de datos

Los siguientes pasos requieren que se copie el archivo tesis.bak al directorio de backups del SQL Server que por defecto es **C:\Archivos de programa\Microsoft SQL Server\MSSQL\BACKUP.**

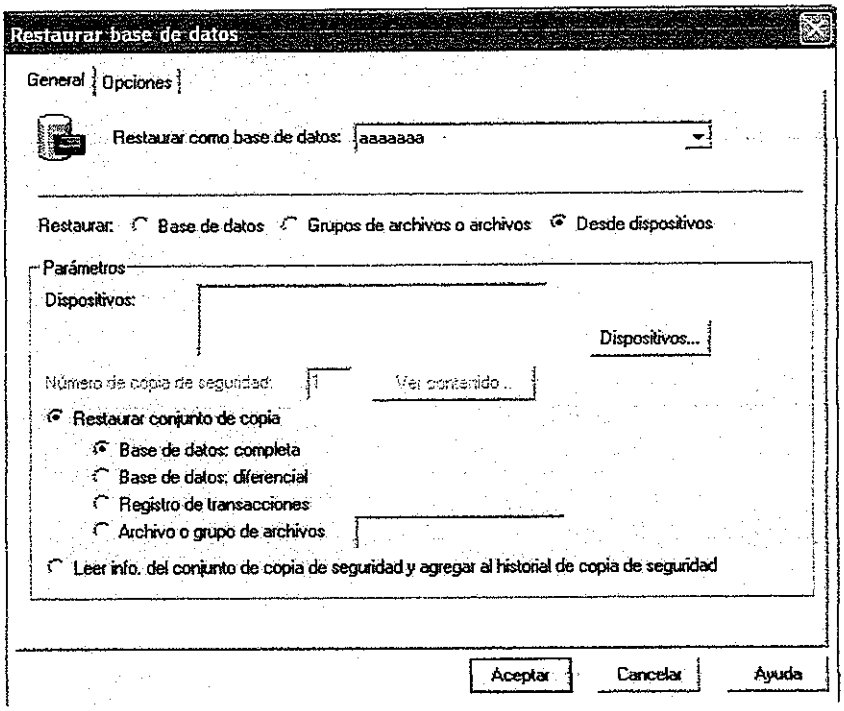
- Iniciar el Administrador Corporativo de SQL Server
- Revisar la conexión con el servidor local
- Crear una base de datos nueva haciendo clic derecho sobre la opción

Base de datos

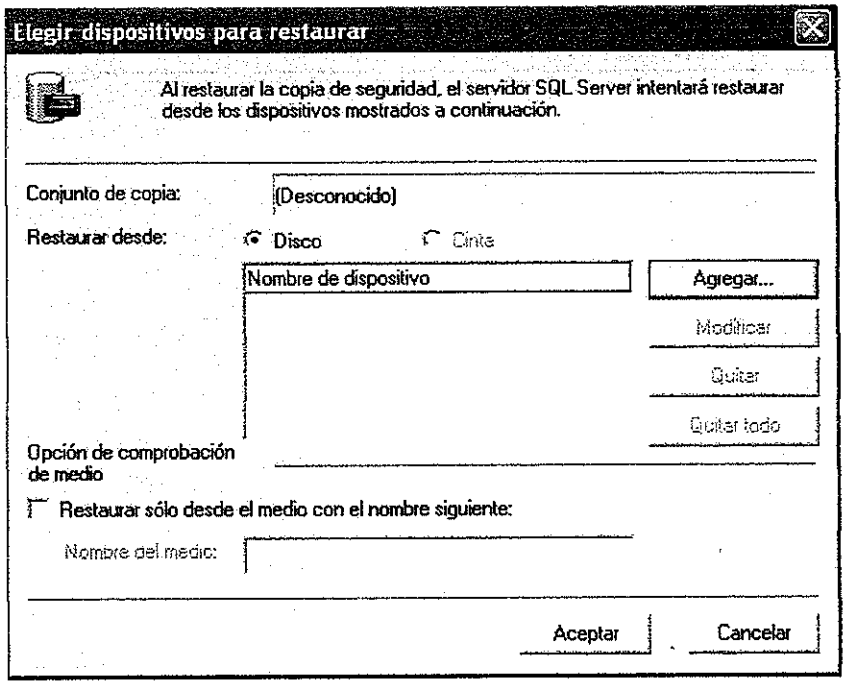
A continuación aparecerá el siguiente cuadro de diálogo, escriba un nombre para la base de datos que va a restaurar.



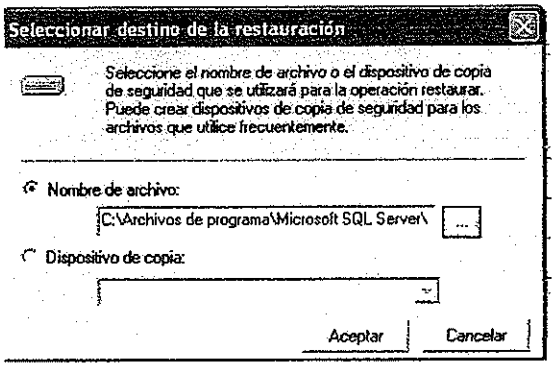
- Luego de crear la base de datos y tenerla vacía, hacemos clic derecho con el Mouse sobre la base de datos nueva y escogemos la opción **Todas las tareas - Restaurar base de datos..**
- Luego aparecerá un cuadro de dialogo como el mostrado a continuación, en el cual tendremos que señalar la opción de Restaurar **desde dispositivo**. En la opción restaurar conjunto de copia escogemos **Bases de datos completa**.



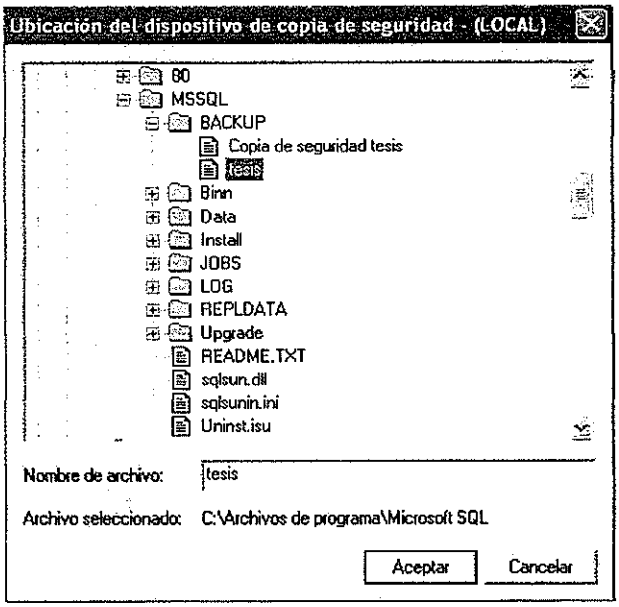
- Luego hacemos click sobre el botón de **Dispositivos** y aparecerá el siguiente cuadro de diálogo:



- En este cuadro de diálogo hacemos clic sobre el botón **Agregar** y aparecerá el siguiente cuadro:



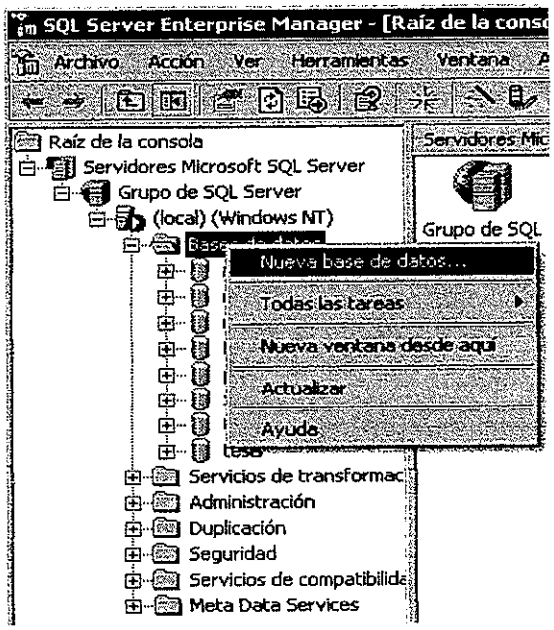
- Luego seleccionamos nuestra copia que pusimos en la carpeta BACKUP, haciendo click en el botón de explorador (...) y podremos visualizar nuestra copia de la siguiente manera:



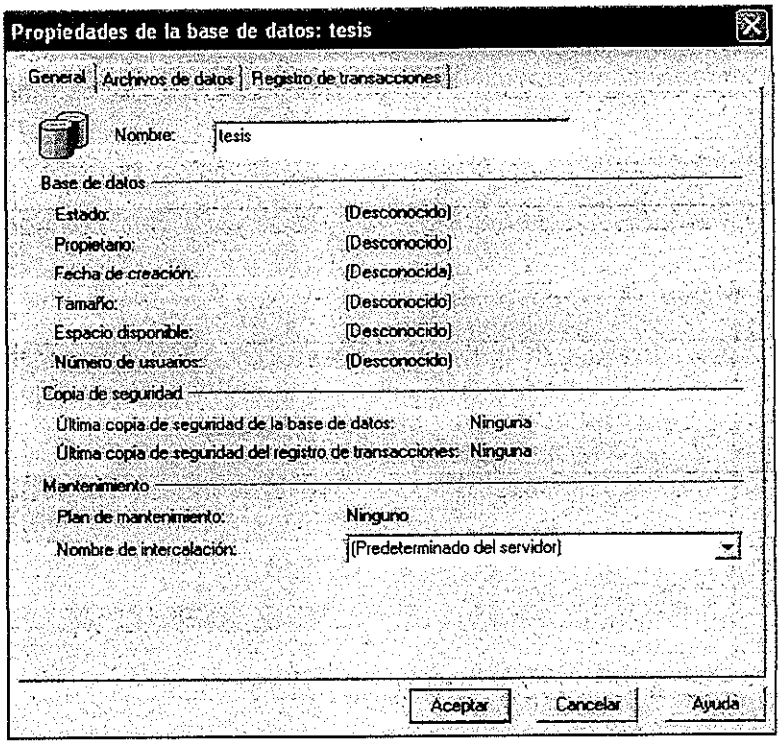
- Finalmente ponemos **Aceptar** en todas los siguientes cuadros de diálogo y **listo.**

Script SQL de la Base de Datos

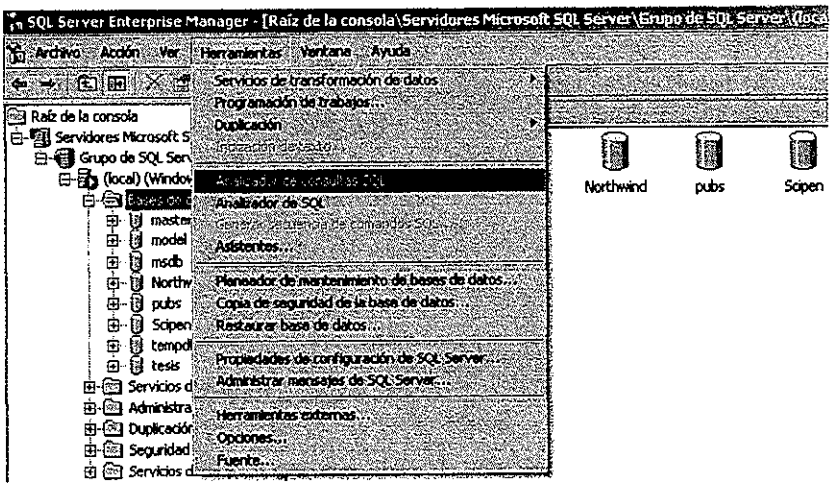
- Pulsar el botón derecho del mouse en la sección de **base de datos** del Administrador Corporativo y seleccione la opción ***Nueva base de datos...***



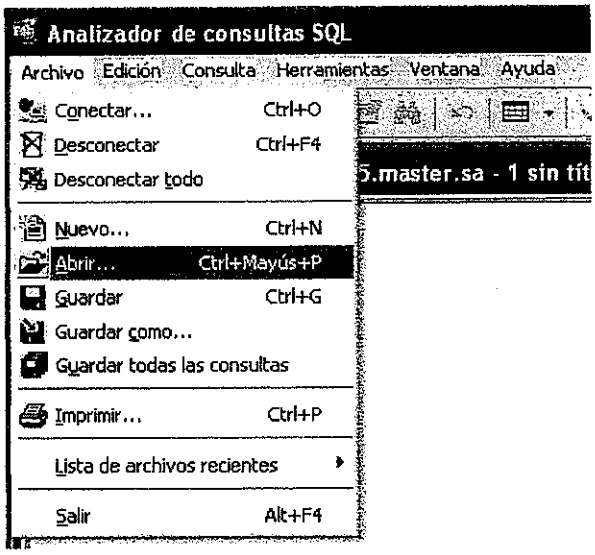
- En el cuadro siguiente dialogo coloque el nombre de la base de datos **tesis** pulse el botón ***OK*** del cuadro de dialogo y se creará una base de datos en blanco (sin datos).



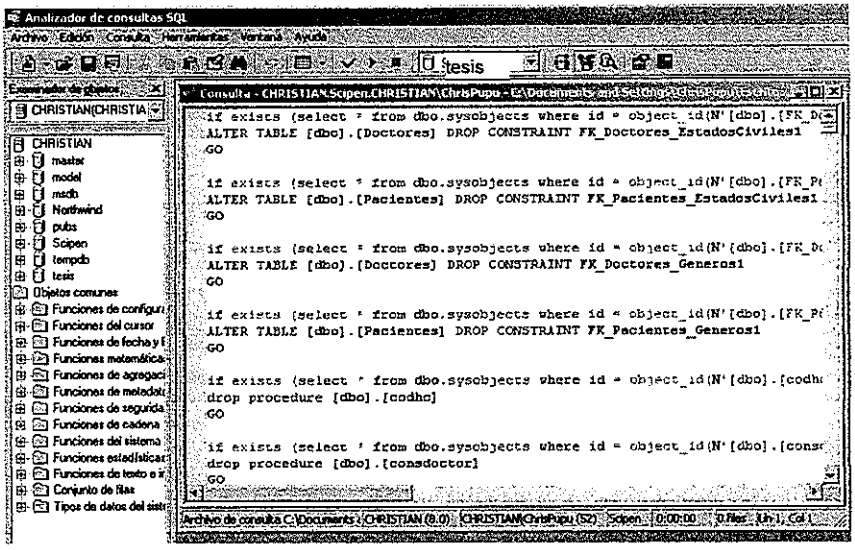
- Luego en el menú ***Herramientas*** seleccione ***Analizador de Consultas SQL.***



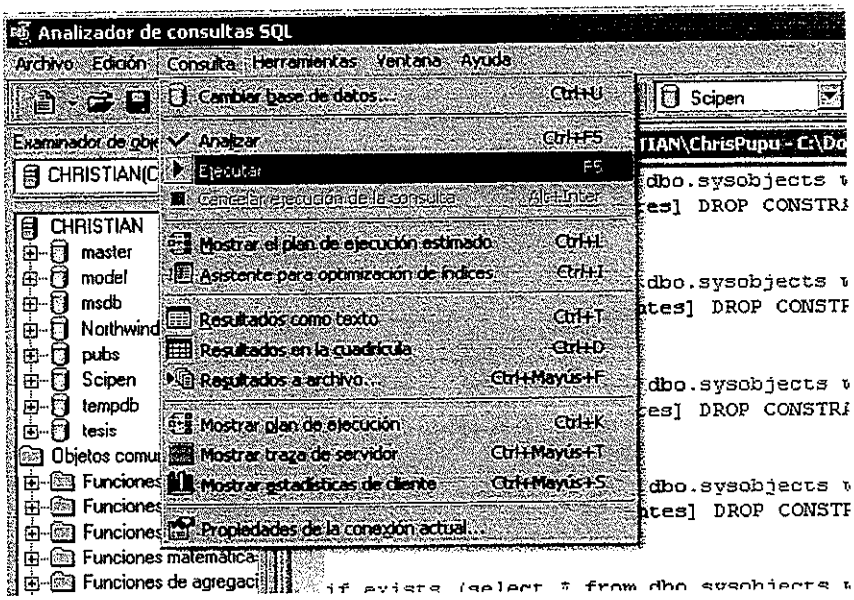
- Aparecerá la ventana del ***Analizador de Consultas SQL*** seleccione el menú **Archivo** seleccione la opción **Abrir.** Abra el archivo **tesis.sql**



- Seleccione la base de datos creada anteriormente



- Seleccione del menú ***Query*** la opción ***Execute*** o pulse ***F5.***



Esto creará la estructura de las tablas dentro de la base de datos.

INSTALACIÓN DE LAS PAGINAS ASP EN EL IIS 5.0 EN WINDOWS XP (PUBLICACIÓN DEL SITIO)

Después de la instalación de la base de datos se deben publicar las páginas de servidor activo (o páginas ASP) en la Intranet. Solo debe copiar los archivos que se encuentran en el CD de instalación bajo el directorio al directorio en el Server ***c:\inetpub\wwwroot.*** Cabe recalcar que se pueden crear subdirectorios dentro de ***c:\inetpub\wwwroot*** y crear un ***directorio virtual*** para un acceso directo al sito. Véase ayuda del IIS 5.0 en windows xp.

INSTALACIÓN DEL ARCHIVO DOTNETCHARTING PARA GRÁFICOS ESTADISTICOS EN PÁGINAS ASP

Para la realización de los gráficos estadísticos se tuvo que instalar los programas: Microsoft.NET.Library y Microsoft.NET.Framework.sdk que son los responsables de la generación de los gráficos que interactúan con páginas ASP creadas en DREAMWEAVER MX. Además se tuvo que crear una carpeta con el nombre **bin** y dentro de esta colocar el archivo dotnetcharting.dll y el dotnetcharting.xml. Para más información visite el sitio www.dotnetcharting.com.

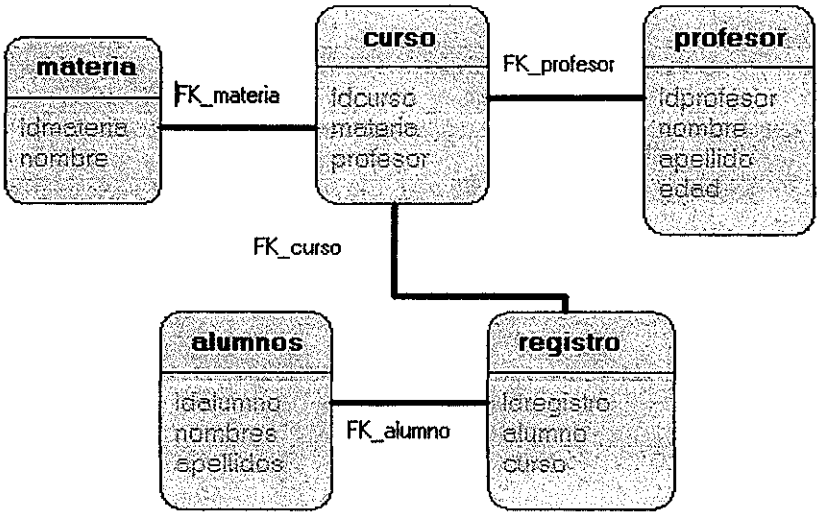
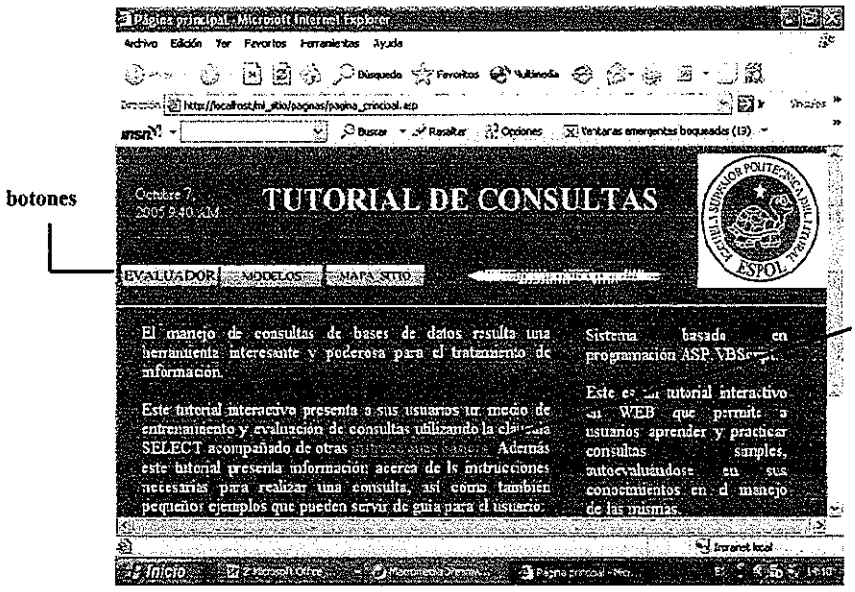


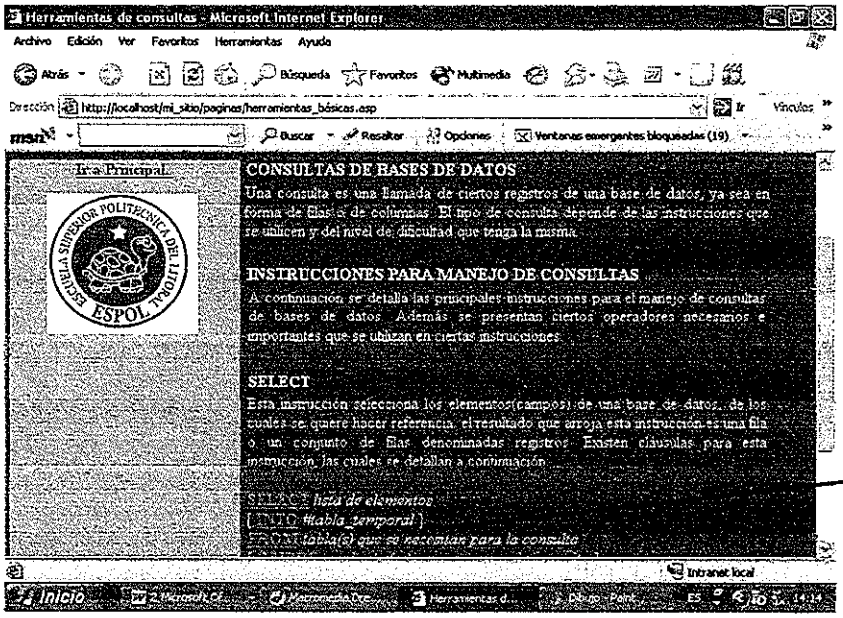
Figura 3

Formularios del tutorial

A continuación se muestra la página principal del tutorial, aquí nos encontramos con una breve descripción del lenguaje en que fue diseñado, un hipervínculo a las sintaxis de las instrucciones básicas para el manejo de las consultas y tres botones que permiten acceder a otras páginas que pertenecen al tutorial. El primer botón es un acceso al formulario del evaluador dinámico, el segundo botón es un acceso al formulario que presenta los diagramas entidad – relación de los modelos de bases de datos que se consideraron para el tutorial, y el tercer botón es un acceso al mapa del sitio, que es sumario de todas las partes que conforman el tutorial.



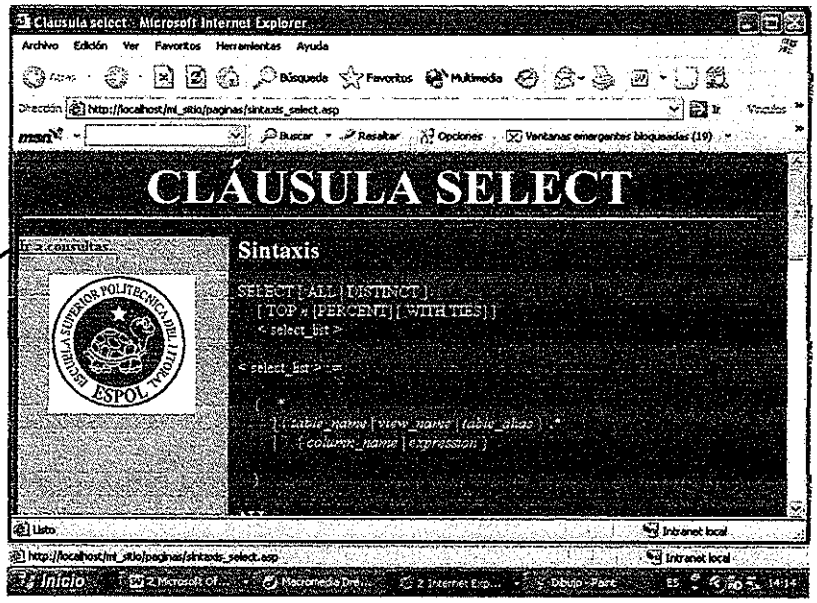
La siguiente página que se presenta, es a la que se accede haciendo click en el hipervínculo de la pantalla principal (**instrucciones básicas**). En este se presentan las diferentes instrucciones básicas par el manejo de consultas, con accesos a otras páginas para cada una de estas instrucciones.



Todas las instrucciones que se presentan en la página anterior, se muestran cada una en interfaces independientes.

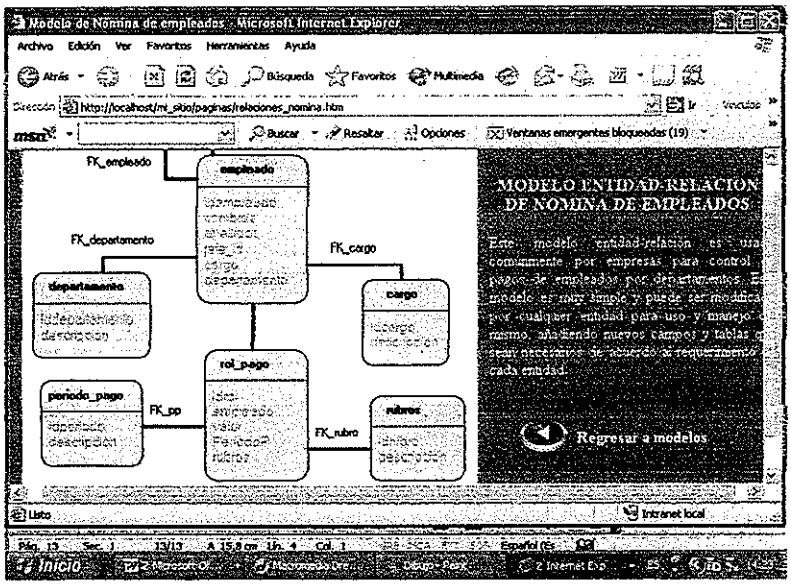
A continuación se presenta la página de la instrucción SELECT. En este se muestra sintaxis, ejemplos y un hipervínculo a la página de las instrucciones básicas.

Hipervínculo a instrucciones básicas

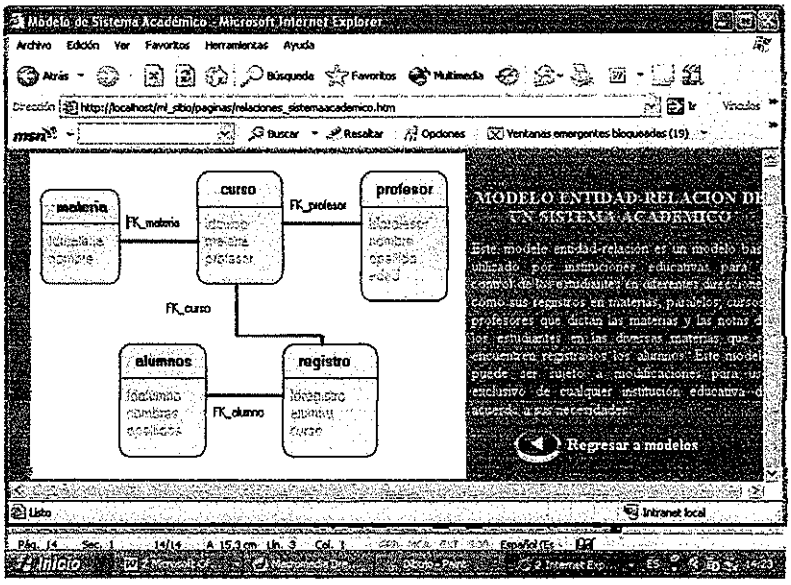


A continuación se presenta la página de la instrucción INTO. En este se muestra sintaxis, ejemplos y un hipervínculo a la página de las instrucciones básicas.

Página que presenta el diagrama entidad – relación de la base nómina de empleados.

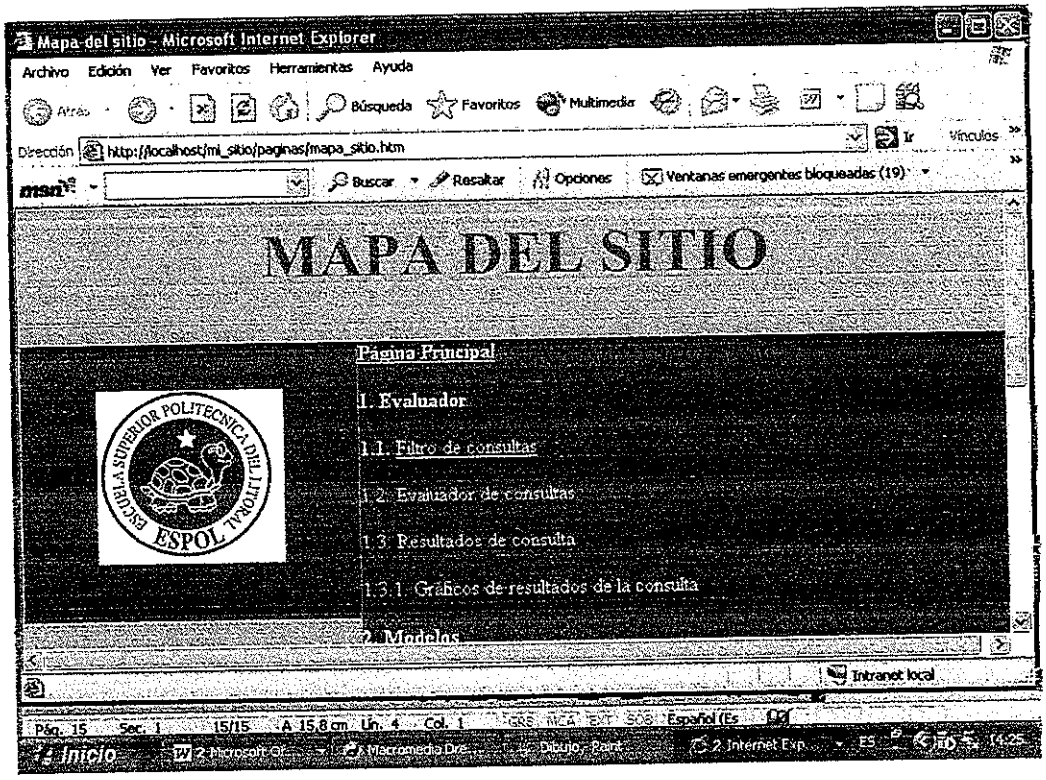


Página que presenta el diagrama entidad – relación de la base nómina de empleados.



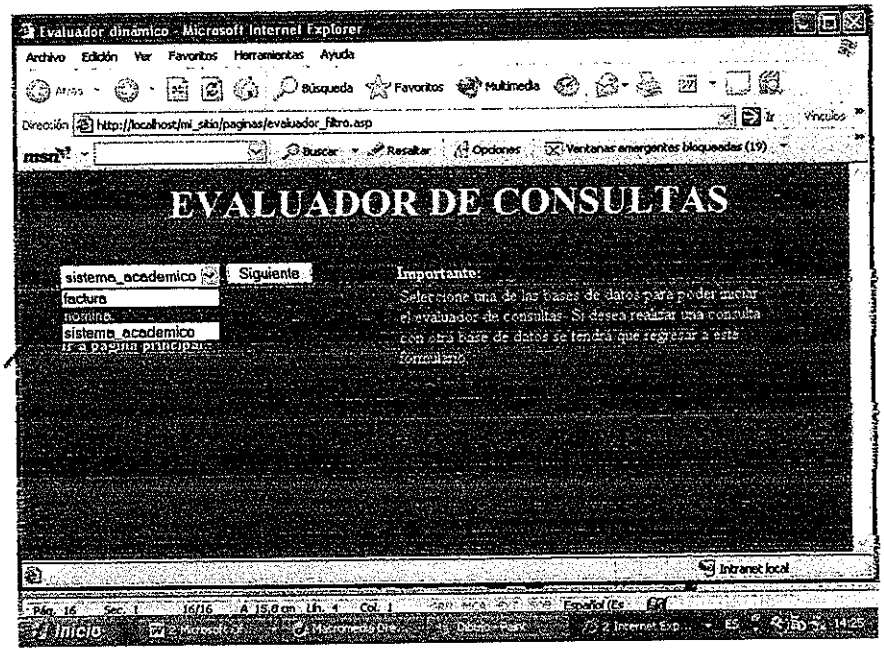
En cada una de las tras páginas presentadas anteriormente encontramos un botón que es un acceso directo a la página que contiene los botones desde donde se accede a los diagramas.

La siguiente página que se presenta es el mapa del sitio. En este encontramos cada uno de los vínculos a todas las páginas que conforman el tutorial.



Existen opciones del mapa que no están como hipervínculos, ya que no pueden ser ejecutadas si antes no se utiliza otra página. Por ejemplo desde el mapa no se puede ejecutar el evaluador de consultas, sin antes ejecutar el filtro de consultas.

A continuación se presenta la página denominada filtro de consultas. En esta página se tiene que escoger la base de datos con la que desea trabajar el usuario. Luego se deberá hacer click en el botón siguiente para que se pueda escribir la sentencia que será evaluada.



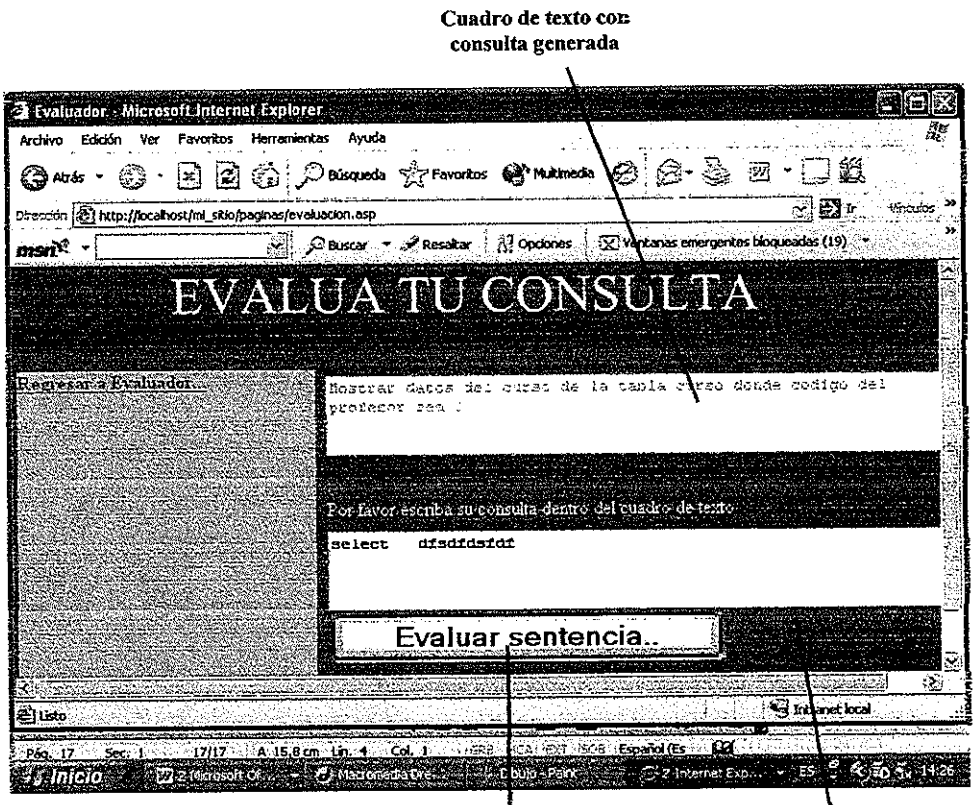
Filtro de la base

Al hacer click en el botón siguiente se carga internamente una consulta con datos de la base que se escogió a través del combo. Esta consulta es presentada en un cuadro de texto que es presentado en la siguiente página.

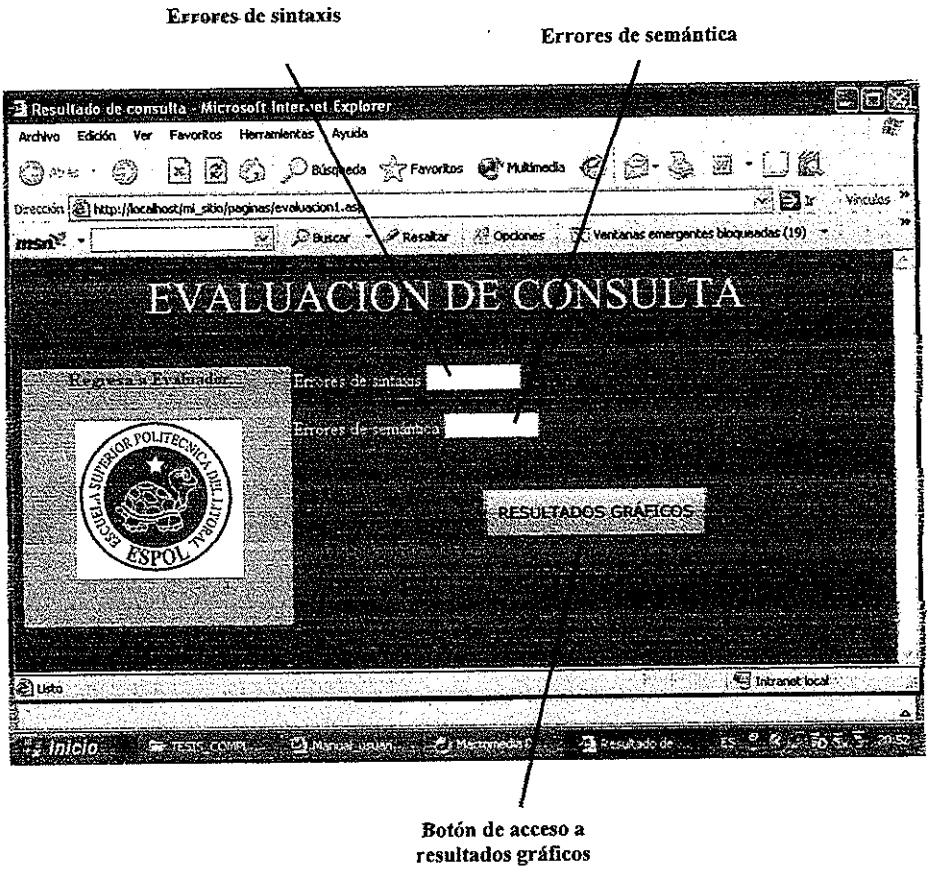
En la página que podemos ver a continuación se presentan dos cuadros de texto. En el primer cuadro se presenta la consulta que se genera internamente y aparece inhabilitado para escritura. En el segundo cuadro se le permite al usuario la escritura de la sentencia que él considere la correcta

para que el compilador diseñado, internamente lo analice y refleje si existen errores. Cabe indicar que el compilador verifica dos tipos de errores: errores de sintaxis y errores de semántica.

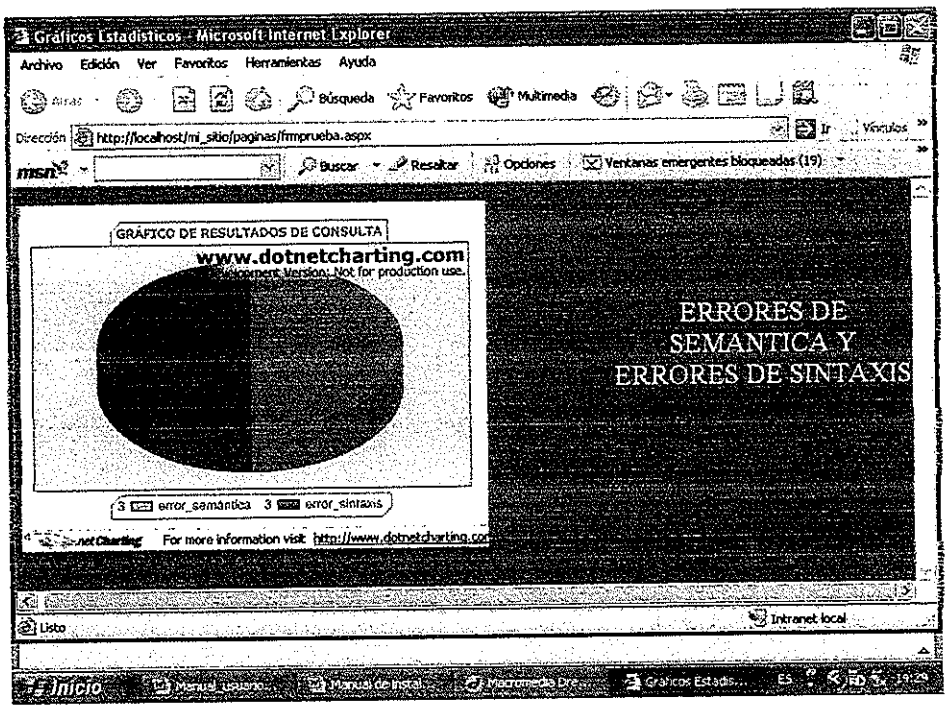
Para que el compilador se ejecute, el usuario luego de escribir su sentencia debe hacer click en el botón que dice **Evaluar sentencia..**



Luego de hacer click en el botón para ejecutar compilador se ejecuta el compilador internamente, se ejecuta y se presenta la siguiente página, en el cual se presentan los tipos de errores que considera el compilador. Además se presenta un botón de acceso directo a una página que me presenta los mismos resultados de manera gráfica.



Luego de presionar el botón **RESULTADOS GRÁFICOS** se carga una página dinámica en la que se presenta un gráfico estadístico de los errores.



La página que se presenta es uno de los gráficos estadísticos que se generan del conjunto de errores que encuentra el compilador.

ANEXO 3

CÓDIGO FUENTE DE LOS PROCEDIMIENTOS EN SQL SERVER

Procedimiento para generación de consulta

```
CREATE procedure pregunta
@bas int
as
--decalración de variables
declare @aleat float
declare @j integer
select @j=id from p
delete from p
set @bas=@j
set @aleat=rand()

--limpieza de las tablas temporal y temporal_d
delete from temporal
delete from temporal_d

--genración de la tabla aleatoria para la consulta
declare @treg int
declare @aleatorio int
declare @maximob int
select @treg=count(Idtabla) from tabla, base where base_id=Idcodigo and
base_id=@bas
select @maximob=max(Idtabla) from tabla where base_id=@bas
set @aleatorio=floor(@maximob-(@treg*rand())+1)
select * from tabla where idtabla=@aleatorio

--generación de la columna aleatoria para la consulta
declare @treg1 int
declare @aleatorio1 int
declare @maximo int
select @treg1=count(Id_columna) from columnas where tabla_id=@aleatorio
select @maximo=max(Id_columna) from columnas where
tabla_id=@aleatorio
```

```
set @aleatorio1=floor(@maximo-(@treg1*rand())+1)
select * from columnas where Id_columna=@aleatorio1

--generación de los valores para la columna de la consulta
declare @treg2 int
declare @aleatorio2 int
declare @m2 int
select @treg2=count(idvalor) from valores where columna_id=@aleatorio1
select @m2=max(Idvalor) from valores where columna_id=@aleatorio1
set @aleatorio2=floor(@m2-(@treg2*rand())+1)
select * from valores where Idvalor=@aleatorio2

--inserción en la tabla temporal de los códigos necesarios para la consulta
insert into temporal values(@bas, @aleatorio, @aleatorio1, @aleatorio2, @aleatorio,0)

--generación de la tabla que sirve de join para la consulta
declare @base nvarchar(20)
declare @tabla1 nvarchar(30)
declare @tabla2 nvarchar(30)
declare @columna nvarchar(30)
declare @valor nvarchar(30)
declare @join nvarchar(30)
select @base=descripcion from base where idcodigo=@bas
select @tabla2=nombre, @tabla1=descripcion from tabla where Idtabla=@aleatorio
select @columna=descripcion from columnas where Id_columna=@aleatorio1
select @valor=valor from valores where Idvalor=@aleatorio2
insert into temporal_d values(@base,@tabla2,@tabla1,@columna,@valor,@tabla1,'ninguna')

--verificación de que un tipo de consultas se ejecuta el 10% de las veces
if @aleat<=0.1 begin
 delete from thijo
 declare @min integer, @max integer,@aleathijo integer
 insert into thijo (idhijo)
  select tabla_hijo_id from refer_f_k where @aleatorio=tabla_padre_id
  select  @min=min(id) from thijo
  select @max=max(id) from thijo
  set @aleathijo=floor((@max-@min)*rand())+@min
  declare @v int
  select @v=idhijo from thijo where id=@aleathijo
```

```
 declare @n nvarchar(50)
 select @n=nombre from tabla where idtabla=@v
 update temporal_d set hijo=@n
update temporal set hijo=@v
end
select * from temporal
select * from temporal_d
```

Procedimiento para revisar sentencia (compilador)

```
CREATE PROCEDURE compilador_m
@t nvarchar(100)
 AS
delete from consulta

--inserción del texto enviado por la pagina web a una tabla estática
declare @qq nvarchar(100)
select @qq=tt from aux_compilador
delete from aux_compilador
set @t=@qq
insert into consulta values(@t)
declare @texto nvarchar(100)
select @texto=ltrim(texto) from consulta
declare @e_sintaxis int
declare @e_semantica int
set @e_sintaxis=0
set @e_semantica=0
```

CIB -ESPOL

```
--contador para consultas de una tabla
declare @t_hijo nvarchar(7)
select @t_hijo=hijo  from temporal_d

--las banderas tienen un valor de 0 sino encuentran lo especificado
        declare @bandera1 int
        select @bandera1=charindex('select ',ltrim(texto),1) from consulta
        if @bandera1=0 begin
                        set @e_sintaxis=@e_sintaxis+1
                end
        if @bandera1<>1 and @bandera1<>0 begin
                        set @e_sintaxis=@e_sintaxis+1
                end

        declare @bandera2 int
        select @bandera2=charindex('from',ltrim(texto),1) from consulta
```

```
        if @bandera2=0 begin
                        set @e_sintaxis=@e_sintaxis+1
                end

        declare @bandera3 int
        select @bandera3=charindex('where', ltrim(texto),1) from consulta
        if @bandera3=0 begin
                        set @e_sintaxis=@e_sintaxis+1
        end
        if @bandera3<@bandera2 begin
                        set @e_sintaxis=@e_sintaxis+1
         end

--contador del error en el primer campo
        declare @columna_temp nvarchar(12)
        select @columna_temp=nombre from temporal, columnas where
        temporal.columna=columnas.Id_columna
        declare @cant_caracteres1 int
        set @cant_caracteres1=len(@columna_temp)
        declare @campo nvarchar(15)
        declare @texto_aux nvarchar(15)
        declare @pos1_espacio int
        declare @pos1_caracter int
        select @pos1_espacio=charindex(' ',ltrim(texto)) from consulta
        set
        texto_aux=substring(@texto,@pos1_espacio,@cant_caracteres1+1)
        set @campo=ltrim(@texto_aux)
        set @pos1_caracter=@pos1_espacio+1
        if @pos1_caracter>@bandera2 begin
                        set @e_sintaxis=@e_sintaxis+1
                end
        if @campo<>@columna_temp
          begin
                set @e_semantica=@e_semantica+1
           end
if @t_hijo='ninguna' begin

        --contador de error de la tabla
        declare @ntabla_temp nvarchar(15)
        declare @bandera_tabla int
        select @ntabla_temp=nombre from temporal,tabla where
        temporal.tabla=Idtabla
        select @bandera_tabla=charindex(@ntabla_temp,ltrim(texto),1) from
        consulta
        if @bandera_tabla=0 begin
```

```
set @e_semantica=@e_semantica+1
        end
if @bandera_tabla<@bandera2 begin
                        set @e_sintaxis=@e_sintaxis+1
end

--contador de error de la condición
declare @conca_condicion nvarchar(30)
declare @nombre_campo nvarchar(15)
declare @valor_campo nvarchar(15)
select @nombre_campo=nombre from temporal, columnas where
temporal.columna=columnas.Id_columna
select @valor_campo=valores.valor from temporal, columnas, valores
where temporal.columna=columnas.Id_columna and
valores.idvalor=temporal.valor
set @conca_condicion=@nombre_campo+'='+@valor_campo
declare @bandera_condicion int
select bandera_condicion=charindex(@conca_condicion,ltrim(texto),1)
from consulta
if @bandera_condicion=0 begin
        set @e_semantica=@e_semantica+1
    end
if @bandera_condicion<@bandera3 begin
                        set @e_sintaxis=@e_sintaxis+1
                end
end
else

--caso contrario (consulta con 2 tablas)
begin

 --contador de error de la tabla
declare @ntabla_temp1 nvarchar(15)
declare @bandera_tabla1 int
select @ntabla_temp1=nombre from temporal,tabla where
temporal.tabla=Idtabla
select @bandera_tabla1=charindex(@ntabla_temp1,ltrim(texto),1) from
consulta
if @bandera_tabla1=0 begin
        set @e_semantica=@e_semantica+1
    end
if @bandera_tabla1<@bandera2 begin
                        set @e_sintaxis=@e_sintaxis+1
                end
```

```
declare @ntabla_temp2 nvarchar(15)
declare @bandera_tabla2 int
select @ntabla_temp2=nombre from temporal,tabla where
temporal.hijo=ldtabla
select @bandera_tabla2=charindex(@ntabla_temp2,ltrim(texto),1)
from consulta
if @bandera_tabla2=0 begin
        set @e_semantica=@e_semantica+1
    end
if @bandera_tabla2<@bandera2 begin
        set @e_sintaxis=@e_sintaxis+1
    end
declare @concatenacion1 nvarchar(30)
declare @concatenacion2 nvarchar(30)
set @concatenacion1=@ntabla_temp1+','+' '+@ntabla_temp2
set @concatenacion2=@ntabla_temp2+','+' '+@ntabla_temp1
declare @v_concatenacion1 int
declare @v_concatenacion2 int
select @v_concatenacion1=charindex(@concatenacion1,texto,1) from
consulta
select @v_concatenacion2=charindex(@concatenacion2,texto,1) from
consulta
if @v_concatenacion1=0 and @v_concatenacion2=0 begin
    set e_sintaxis=@e_sintaxis+1
end

--contador de error de la condición
declare @conca_condicion1 nvarchar(30)
declare @nombre_campo1 nvarchar(15)
declare @valor_campo1 nvarchar(15)
select @nombre_campo1=nombre from temporal, columnas where
temporal.columna=columnas.ld_columna
select @valor_campo1=valores.valor from temporal, columnas, valores
where temporal.columna=columnas.ld_columna and
valores.idvalor=temporal.valor
set @conca_condicion1=@nombre_campo+'='+@valor_campo
declare @bandera_condicion1 int
select @bandera_condicion1=charindex(@conca_condicion,ltrim(texto),1)
from consulta
if @bandera_condicion1=0 begin
        set @e_semantica=@e_semantica+1
    end
if @bandera_condicion1<@bandera3 begin
        set @e_sintaxis=@e_sintaxis+1
    end
```

```
        end

set @e_sintaxis=@e_sintaxis*2
set @e_semantica=@e_semantica*1
declare @visitante int
select @visitante=max(nombreid) from cabecera_visita
--delete from c_errores
insert into c_errores values(@visitante,2, @e_sintaxis)
insert into c_errores values(@visitante,1, @e_semantica)
exec numero_errores
exec errores_x_iteracion
select * from error_iteracion
```

Procedimiento para almacenar datos del usuario

```
CREATE PROCEDURE calificacion
as
delete from calificacion_consulta
declare @a int
select @a=(error_sintaxis+error_semantica) from error_iteracion
if @a=0 begin
        insert into calificacion_consulta values(100)
end
if @a>=1 and @a<=3 begin
        insert into calificacion_consulta values(90)
end
if @a>=4 and @a<=6 begin
        insert into calificacion_consulta values(70)
end
if @a>=7 and @a<=9 begin
        insert into calificacion_consulta values(60)
end
if @a>=10 and @a<=15 begin
        insert into calificacion_consulta values(20)
end
if @a>=16 and @a<20 begin
        insert into calificacion_consulta values(0)
end
```

BIBLIOGRAFÍA

1. ABRIL, G. : "Sujetos, interfaces, texturas", en Revista de Occidente, número 206, Madrid, 1998.
2. Cfr. BOTKIN J., Aprender, horizonte sin límites. Informe al Club de Roma, Editorial Santillana, Madrid, 1979.
3. ESCUDERO, J.M. : "La integración escolar de las nuevas tecnologías de la información", en Infodidac, número 21, Barcelona, 1992.
4. FRIED SCHNITMAN, D.: Nuevos paradigmas, cultura y subjetividad, Buenos Aires : Paidós, 1994.
5. LANDOW, G.P.: Hipertexto. La convergencia de la teoría crítica contemporánea y la tecnología , Barcelona : Paidós, 1995
6. MASTERMAN, L.: La enseñanza de los medios de comunicación, Madrid : Ediciones de la Torre, 1993
7. Cfr. SIERRA F., El futuro de la civilización tecnológica. Utopías, distopías y entropías en la era de la telaraña electrónica, Universidad del Mayab, México, 1996.
8. TRABER, M. (Ed.): The myth of Information Revolution, California : Sage, 1988.