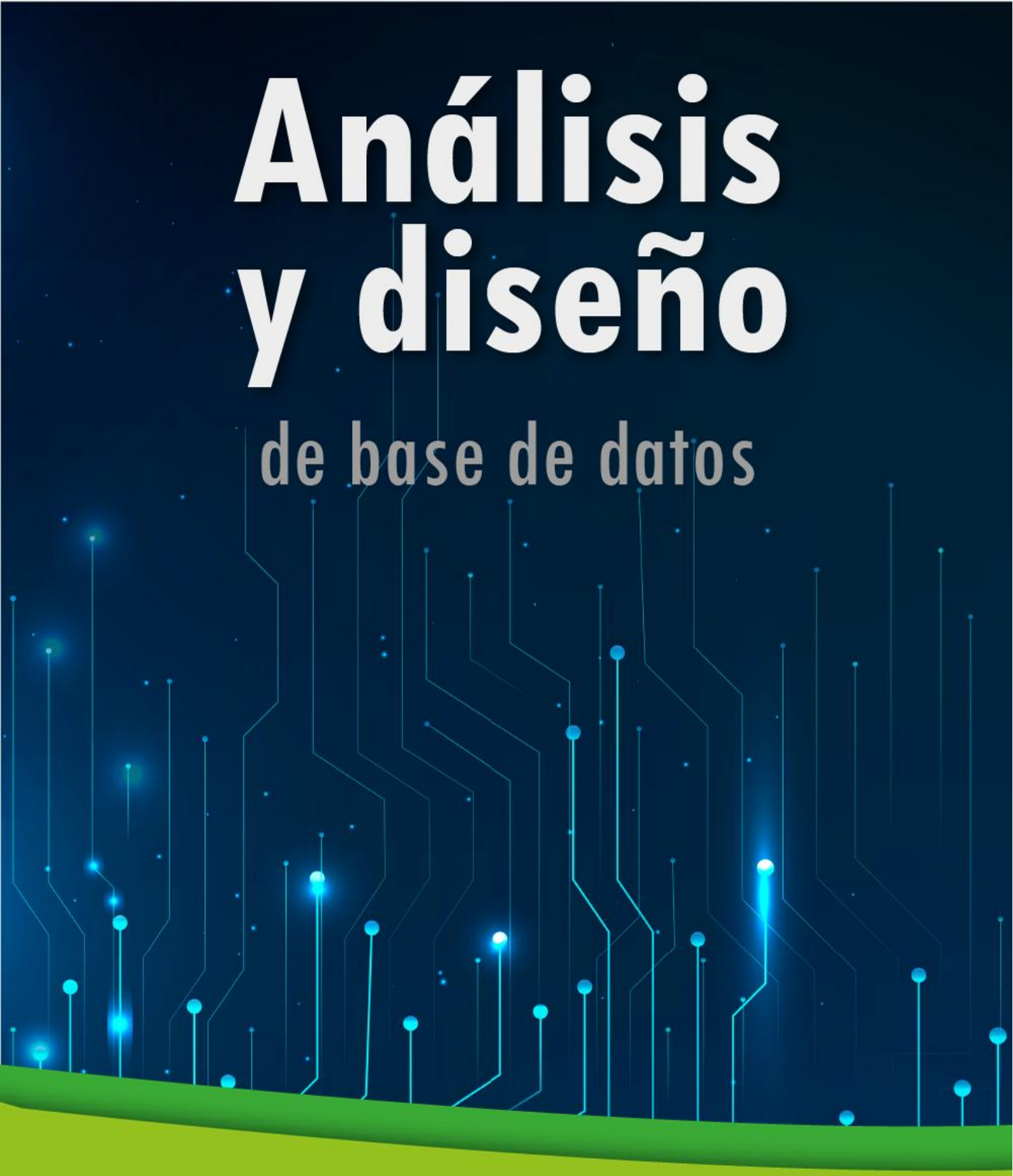
Análisis y diseño

de base de datos



Robert Wilfrido Moreira Centeno
Edison Ernesto Almeida Zambrano
Homero Renán Mendoza Rodríguez
Esthela María San Andrés Laz
Katty Tatiana Mendoza Muñoz

Uleam
Editorial Universitaria

Análisis y diseño de base de datos

Robert Wilfrido Moreira Centeno

Edison Ernesto Almeida Zambrano

Homero Renán Mendoza Rodríguez

Esthela María San Andrés Laz

Katty Tatiana Mendoza Muñoz

Universidad Laica Eloy Alfaro de Manabí
Ciudadela universitaria vía circunvalación (Manta)
www.uleam.edu.ec

Dr. Marcos Zambrano Zambrano, PhD
Rector
Dra. Jackeline Terranova Ruiz, PhD
Vicerrectora de Investigación, Vinculación y Postgrado
Dr. Pedro Quijije, PhD
Vicerrector Académico
Dr. Fidel Chiriboga Mendoza, PhD
Director de Editorial Universitaria

Análisis y diseño de base de datos

Robert Wilfrido Moreira Centeno
Edison Ernesto Almeida Zambrano
Homero Renán Mendoza Rodríguez
Esthela María San Andrés Laz
Katty Tatiana Mendoza Muñoz

Edición: Primera. Julio 2022. Publicación digital

ISBN: 978-9942-827-76-0

Editorial Universitaria
Mg. Alexis Cuzme Espinales
Editor General
Mg. José Márquez Rodríguez
Gestor de Diseño Editorial
Mg. Rossana Cedeño García
Gestora de Redacción y trámites documentales del editorial con los autores.
Lic. Anyela Rivas Cevallos
Secretaria General de la Editorial

Una producción de la Universidad Laica Eloy Alfaro de Manabí, registrada en la Cámara Ecuatoriana del Libro.

Sitio Web: www.munayi.uleam.edu.ec
Correo institucional: editorial@uleam.edu.ec
Facebook @Ediciones Uleam
Twitter @EdicionesUleam
Teléfono: 2 623 026 Ext. 255

A Dios y mi familia por siempre su valioso apoyo.

Robert

Al creador supremo y a todos quienes han aportado siendo parte de mi vida o dejándome formar parte de ella.

Edison

A Dios, a todas y todos quienes aportan significado para desarrollar esfuerzos en bien de la comunidad universitaria.

Homero

A Dios, a mi madre y a toda mi familia que son la razón de mi vida.

Esthela

A todos quienes fueron parte de este proyecto.

Katty

Resumen

La información en el contexto empresarial se constituye como un recurso valioso, pues en los niveles siguientes se transforma en conocimiento y luego en decisiones que permitirán a una organización mantenerse operativa y transparente para los niveles administrativos y operativos.

El propósito del conocimiento y las decisiones son alcanzados por las bases de datos, mismas que en general hacen referencia a un conjunto de datos multidimensionales en el sentido de que los enlaces internos existentes entre los distintos elementos hacen que se pueda acceder a la información con diversas perspectivas, siempre conservando la integridad y la consistencia dentro de un modelo de datos particular.

Las bases de datos relacionales como evolución de los sistemas de archivos, modelos de red y jerárquicos propiciaron una disciplina en la que se invierten muchos esfuerzos de investigación, ya que ir desde el dato hasta las decisiones en una empresa implica un ciclo de gestión que va desde el desarrollo, implantación, mantenimiento y evolución.

Se abordará en las siguientes páginas el Análisis y Diseño de Bases de Datos como solución al hallazgo de datos no estructurados y semiestructurados de una empresa que propicia con su redundancia y falta de orden, excesivo trabajo a la hora de requerir resultados rápidos de búsqueda, así como responder a contingencias naturales o empresariales.

El correcto diseño de una base de datos hará gestionable para la administración las inversiones necesarias de recursos para mantener en niveles de funcionamiento óptimo en las partes esenciales que apoyan al sistema de base de datos en general como lo son el hardware, software, personas, procedimientos y datos.

Palabras clave: Bases de datos, sistemas de bases de datos, gestión, análisis, diseño, integridad, consistencia, decisiones, conocimiento, modelos de datos, modelo relacional, datos estructurados, datos semiestructurados, componentes de sistemas de bases de datos, conceptos bases de datos.

CONVENCIONES TIPOGRÁFICAS

Se ha adoptado un conjunto de convenciones tipográficas con el objetivo de mejorar la lectura de los conceptos y comprender los ejercicios planteados.

Los estilos empleados en el texto son:

- El texto general del libro se utiliza la letra Times New Roman.
- Las palabras reservadas se destacan con color morado. Ejemplo: new, for, if, etc.
- Los nombres de los programas, clase, métodos se destacan con tipo de letra Courier (11 puntos). Ejemplo: `Estudiante`, `mostraMensaje`, `IPersona`, etc.
- Toda palabra que no sea en español se destaca con cursiva. Ejemplo: *String Development Kit*, *swing*

Los estilos empleados en los ejemplos son:9999

Se han omitido las tildes y eñes en los identificadores de los ejemplos propuestos al ser Java un lenguaje escrito en inglés. Ejemplo: anio, dias, numero, etc.

Los ejemplos normalmente están completos y por lo tanto se pueden escribir y probar. Se los ha encerrado en cajas y están numerados. El texto de los ejemplos emplea un tipo de letra de paso fijo Courier. Ejemplo:

Ejemplo 1 Estilo de código en base de datos

```
1   CREATE TABLE auditoria
2    (
3     id serial NOT NULL,
4     "NombreTabla" character(45) NOT NULL,
5     "Operacion" char(1) NOT NULL,
6     "ValorAnterior" text,
7     "ValorNuevo" text,
8     "Fecha" timestamp without time zone NOT NULL,
9     "Usuario" character(45) NOT NULL,
10    CONSTRAINT pk_auditoria PRIMARY KEY (id)
11  );
```

- Los números de línea permiten hacer referencia a una instrucción concreta del código.
- Los ejemplos parciales tienen el mismo formato que un ejemplo completo, pero no se colocará su título.

- La salida que genera un código de ejemplo o un programa se encierra en un cuadro y están numerados (el texto de las salidas emplea un tipo de letra de paso fijo Courier). Ejemplo:

```
1  Ingrese el primer número
2  7
3  Ingrese el segundo número
4  8
5  Ingrese el tercer número
6  6
7  El número mayor es = 8
```

- Las recomendaciones o sugerencias se mostrarán a través de un cuadro sin numeración. Ejemplo:

Recomendación
El modelo relacional es lo que se conoce como modelo lógico de bases de datos y el esquema conceptual es lo que se conoce como modelo conceptual.

Contenido

Índice de figuras

Índice de tablas

Introducción

El presente trabajo se sustenta en años de experiencia en la enseñanza y aprendizaje de bases de datos, un término que resulta común para quienes se forman en el perfil profesional de tecnologías de la información, y se constituye en un eje fundamental para los trabajos desarrollados en esta área.

Se aborda el problema de los datos no estructurados en que, al no tener formato específico, su estructura normalmente no es uniforme y se tiene poco o nulo control sobre ellos, la información no está representada por datos elementales y su interpretación y manipulación es mucho más compleja como lo es el caso de los archivos pdf, correo electrónico, música, archivos JPG, videos, documento de texto.

Los sistemas de bases de datos que cohesionan hardware, software, redes, personal y políticas no tienen un momento en concreto en el cual hayan surgido con respecto a los sistemas de ficheros que fueron en su tiempo la mejor opción para la forma manual de organizar y manejar datos, pero estos últimos se estudian para dar ideas de las ventajas que tienen las bases de datos relacionales con respecto a ellos.

Los modelos son abstractos, comprensibles, precisos, predictivos y baratos, pero nunca completos, en virtud de ello nace el concepto de modelo de datos, dentro de ese conjunto destaca aún con mucha fuerza el modelo de datos relacional que se concibe en la matriz de un sistema gestor de base de datos como el modelo lógico que da lugar al modelo físico que propicia integridad y consistencia a la información guardada en una empresa.

Notaciones tradicionales como la de Chen del modelo entidad relación que destaca como el modelo conceptual de nivel pedagógico suficiente y rico permite abordar el concepto de entidad, relación o asociación y atributos que permitirán moldear un conjunto de requisitos denominados dominio del universo.

Se hace distinción de cuál es el alcance real de una base de datos y que tipo de software permite implementarlo, siempre con la condición del soporte para auto documentación a través de metadatos, imposición de dominio de datos, relaciones definidas entre tablas, todas estas características que permiten la capacidad de navegación con multi perspectiva hacia los datos.

Se aclaran un término valioso como el de reglas de negocios, mismo que, aunque en su definición no abarcan solo a negocio sino a organizaciones en general se constituye en el eje fundamental para entender las política, procedimientos o

normas adoptadas por una organización, y que sirven en general para definir entidades, atributos, relaciones y restricciones de un modelo de datos.

Este trabajo explicará el modelo de datos relacional en función de sus características que lo distinguen como lo es: presentar al usuario los datos en forma de tabla, distinguir los sistemas de base de datos relacional de los sistemas de archivos y de los sistemas anteriores de bases de datos tipo red a través de su estructura de relación, y aceptar un lenguaje de consulta set-at-a time especificando qué hacer sin decir cómo hacerlo.

El proceso de modelar datos es iterativo y progresivo, ayuda a que el lector entienda las complejidades del ambiente real, a través de herramientas conceptuales que permiten describir los datos, sus relaciones, límites de integridad que les afectan, así como la terminología a emplear.

Luego de estudiado todos los principios del modelo relacional un apartado fundamental en este libro es el lenguaje de consulta que permite obtener respuestas rápidas a las necesidades de información de una empresa.

Esperamos que el resto del material cubra con varias de las inquietudes que tienen aquellos que se encuentran con términos que muchas veces dificulta la interpretación de todos los temas de bases de datos relacionales, se abordaran conceptos usando diversos gestores de bases de datos los cuales en sus principios respetan al modelo relacional y que, sin embargo, difieren en su sintaxis a nivel del modelo físico que es SQL.

1 SISTEMAS DE INFORMACIÓN Y BASES DE DATOS

En este capítulo se explica el concepto de la información como un conjunto de elementos, procedimientos y acciones, interrelacionados entre sí y que cuyo fin es la producción de información confiable y veraz que apoye la toma de decisiones necesarias para la marcha y el control de la organización de la que forma parte integrante.

Se detalla la estructura de la base de datos y los sistemas de base de datos, sus principales conceptos claramente descritos para el entendimiento del lector.

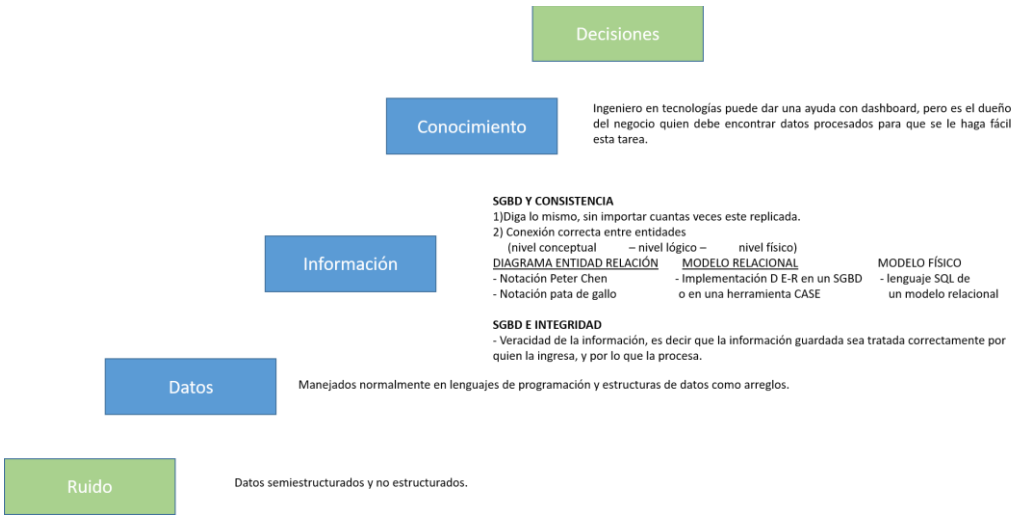
Objetivos:

- Conocer la definición de la información y sistemas de información.
- Identificar la estructura de los sistemas de base de datos.
- Dominar la terminología aceptada de Bases de Datos y Sistemas de Información.

1.1 Sistemas de información y base de datos

La información es el resultado de procesar datos sin elaborar para dejar ver su significado, para revelar dicho significado la información necesita de un contexto, por ejemplo, una lectura promedio de temperatura de 70 grados no significa mucho, a menos que se conozca su contexto: ¿Esto está en grados Fahrenheit o Celsius? ¿Esta temperatura es de una máquina, de un cuerpo o del aire a la intemperie?

Figura 1. Orígenes de un sistema de información

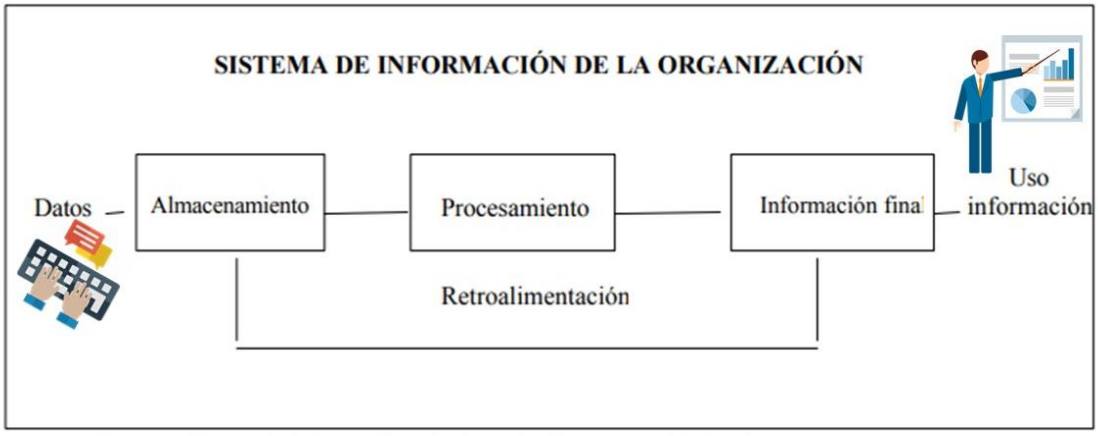


Los sistemas de información son todos aquellos recursos que se organizan para que partiendo del denominado “ruido” consistente en datos estructurados y semiestructurados, se sirva de formas de automatización basadas en comunicación y organización de actividades, que permitirán organizar el trabajo operativo de una organización y dar a los altos directivos formas innovadoras de ver el estado de su empresa para tomar decisiones.

La información oportuna y útil requiere datos precisos manejados en términos de consistencia e integridad como se detalla en la Figura 1, dichos criterios son los lineamientos para que una estructura de datos sea considerada como “base de datos”, los datos guardados deben ser generados y guardados en forma apropiada en un formato que sea fácil de acceder y procesar.

1.1.1 Sistemas de información

Figura 2. Sistemas de Información en una organización



Es un conjunto de componentes interrelacionados que obtiene, procesa, almacena y distribuye información para apoyar la toma de decisiones y el control de una organización, cualquier sistema de información realiza cuatro actividades básicas: entrada, almacenamiento, procesamiento y salida de información.

Entrada de Información:

Es el proceso mediante el cual el Sistema de Información toma los datos que requiere para procesar la información, las entradas de información pueden ser manual o automática. Las manuales son aquellas que se proporcionan en forma

directa por el usuario, mientras que las automáticas son datos o información que provienen o son tornados de otros sistemas o módulos.

Almacenamiento de información:

Es una propiedad del sistema que puede recordar la información que se guarda en las entradas de datos, para luego dirigirlas al caso de almacenamiento lógico en caso de los sistemas gestores de bases de datos y almacenamiento físico en el caso de sistemas de archivos.

Procesamiento de Información:

Es la capacidad de organizar los datos para requerimientos particulares operativos o administrativos, estos pueden ser los recientemente generados o los que tengan cierta antigüedad, el procesamiento debe estar enmarcado en función de las reglas del negocio definidos para un determinado sector.

Salida de Información:

Es la capacidad de un sistema de información para extraer la información procesada o bien datos de entrada al exterior, esta puede ser visualizada o almacenada por medio de dispositivos electrónicos como: las impresoras, terminales, diskettes, USB, etc.

En el corazón de todos estos sistemas están la captura, el almacenamiento, agregado, la manipulación, la diseminación y la administración de datos.

1.1.2 Base de datos (BD)

El término base de datos hace referencia a un conjunto de datos multidimensionales en el sentido de que los enlaces internos existentes entre los distintos elementos hacen que se pueda acceder a la información con diversas perspectivas. Esto difiere de un sistema de archivos tradicional, que es denominado como archivo plano, que no es más que un sistema de almacenamiento unidimensional.

Una base de datos es una estructura computarizada compartida e integrada que guarda un conjunto de:

- Datos del usuario final, es decir, datos sin elaborar que son de interés para el usuario final.
- Metadatos, o datos acerca de datos, por medio de los cuales los datos del usuario final son integrados y manejados.

Los metadatos dan una descripción de las características de los datos y del conjunto de relaciones que enlaza los datos encontrados dentro de la base de datos. Los metadatos documentan entre otras cosas, que tablas existen en un base de datos, las columnas que posee cada una de las tablas, el tipo de datos que puede almacenar, etc.

Figura 3. Bases de datos manejadas independientemente

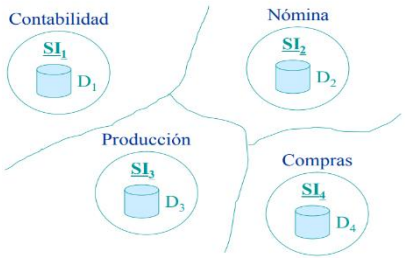
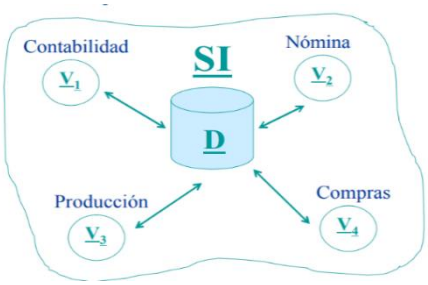


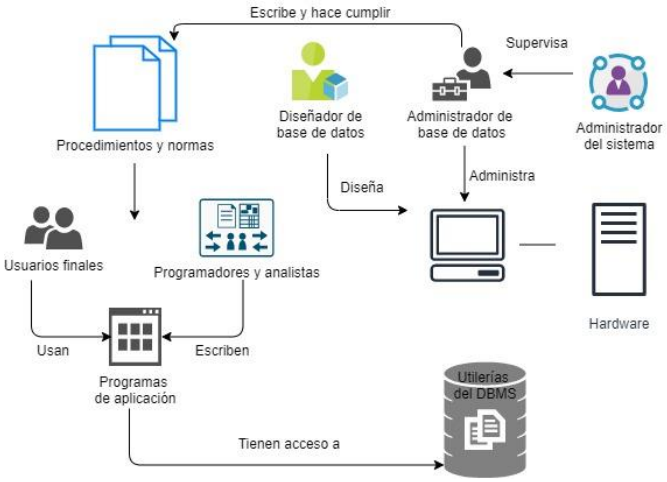
Figura 4. Bases de datos manejadas en un solo conjunto



1.1.3 Sistemas de base de datos

El término sistema de base de datos se refiere a una organización de componentes que define y regula la captura, almacenamiento, administración y uso de datos dentro de un ambiente de base de datos. Desde el punto de vista de una administración general, el sistema de base de datos está compuesto de cinco partes principales *hardware, software*, personas, procedimientos y datos.

Figura 5. El ambiente de un sistema de base de datos (Coronel,2011)



Software

Para que el sistema de base de datos funcione a plenitud se requieren tres tipos de software: el sistema operativo, software el DBMS y programas de aplicación y utilerías. El software del sistema operativo maneja todos los componentes del hardware y hace posible que todo el demás software se ejecute en las computadoras. El software del DBMS maneja la base de datos dentro del sistema de base de datos. Los programas de aplicación y utilerías se usan para tener acceso y manipular datos en el DBMS y manejar el ambiente de computadora en el que tienen lugar el acceso y la manipulación de datos.

Personas

Este componente incluye a todos los usuarios del sistema de base de datos. A partir de las funciones centrales de un trabajo, se pueden identificar cinco tipos de usuario en un sistema de base de datos: administradores del sistema, administradores de la base de datos, diseñadores de la base de datos, analistas y programadores del sistema y, por último, usuarios finales.

Procedimientos

Son las instrucciones y reglas que gobiernan el diseño y uso del sistema de base de datos. Los procedimientos son un elemento crítico del sistema, aun cuando a veces es olvidado. Los procedimientos desempeñan una importante función en una compañía porque hacen cumplir las normas por medio de las cuales se realizan los

negocios dentro de la organización y con sus clientes. Los procedimientos también se emplean para asegurar que hay una forma organizada de vigilar y auditar tanto los datos que entran a la base de datos como la información que se genera con ellos.

Datos

La palabra datos comprende el conjunto de datos almacenados en la base de datos. Como los datos son la materia prima de la que se genera información, la determinación de cuáles han de introducirse en la base y cómo han de organizarse es una parte vital del trabajo del diseñador.

1.4 Reglas de negocios

Desde el punto de vista de una base de datos, el conjunto de datos tiene sentido solo cuando refleja debidamente reglas de negocios. Una regla de negocios es una descripción breve, precisa y no ambigua de una política, procedimiento o principio dentro de una organización específica.

Las reglas de negocio son muy importantes en el diseño de una base de datos porque determinan los controles que deben aplicarse a los datos. Casi todas las reglas de negocios se imponen mediante procedimientos manuales que se ordena seguir a los empleados o mediante la lógica de los programas de aplicación.

Las reglas de negocios debidamente escritas se usan para definir entidades, atributos, relaciones y restricciones. Cada vez que se vean enunciados de relación, como “un agente puede atender a varios clientes y cada cliente puede ser atendido por un solo agente”, se observan reglas de negocios aplicadas. Se verá la aplicación de reglas de negocios en todo este libro, en especial en los capítulos dedicados al modelado de datos y diseño de bases de datos.

Para ser eficientes, las reglas de negocios deben ser fáciles de entender y ampliamente diseminadas, para asegurarse que toda persona dentro de la organización comparta una interpretación común de las reglas. Las reglas de negocios describen, en lenguaje sencillo, las características principales y distintivas de los datos como los ve la compañía. Ejemplos de reglas de negocios son los siguientes:

- Un cliente puede generar muchas facturas.

- Una factura es generada por un solo cliente.
- Una sesión de capacitación no puede ser programada para menos de 10 o para más de 30 empleados.

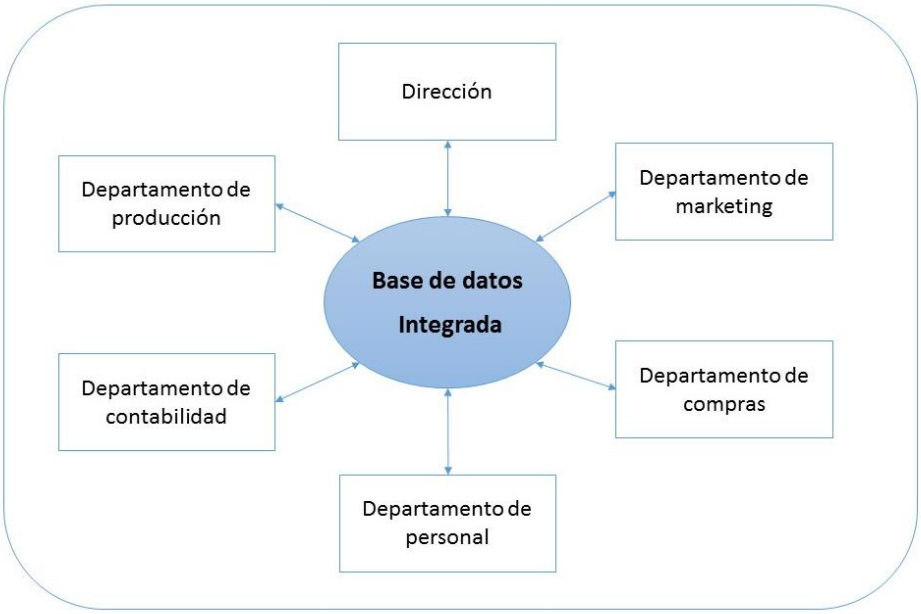
1.2 Sistemas de bases de datos y sus aplicaciones

1.2.1 Importancia de los sistemas de base de datos

Desde los comienzos de las computadoras, estas fueron utilizadas para el campo de la gestión de información, cada aplicación tendía a ser implementada como un conjunto separado con su propio conjunto de datos. Las nóminas se procesaban utilizando el archivo de nóminas, el departamento de personal mantenía sus propios registros de empleados y el inventario se gestionaba mediante un archivo de inventario. Esto significaba que buena parte de la información requerida por una organización estaba duplicada por toda la empresa y que muchos elementos diferentes, pero relacionados, se almacenaban en sistemas separados. En este tipo de entorno, surgieron como medio para integrar la información almacenada y mantenida por una organización.

Estos sistemas de base de datos proporcionan un valioso recurso con el que tomar decisiones de gestión, suponiendo que pudiera accederse a la información de una manera significativa. Hoy en día la tecnología de base de datos, combinadas con las técnicas de minería de datos constituyen una importante herramienta de gestión, permitiendo a la organización extraer información relevante a partir de enormes cantidades de datos que cubren todos los aspectos de la organización y su entorno.

Figura 6. Sistema de información orientado a base de datos.



Las bases de datos es el hilo conductor de las tecnologías de almacenes de datos, procesamiento analítico en línea (OLAP) y *data mining*.

1.3 Sistemas de bases de datos frente a los sistemas de archivos

Los predecesores de los sistemas de bases de datos fueron los sistemas de ficheros o archivos. Un sistema de ficheros está formado por un conjunto de ficheros de datos y los programas de aplicación que permiten a los usuarios finales trabajar sobre los mismos. No hay un momento concreto en el que los sistemas de ficheros hayan cesado y hayan dado comienzo los sistemas de bases de datos. De hecho, todavía existen sistemas de ficheros en uso.

1.3.1 Sistemas de archivos

El método de sistema de archivos para organizar y manejar datos fue una mejora definitiva sobre el sistema manual y cumplió un propósito útil en el manejo de datos durante más de dos décadas, tiempo muy largo en la era de las computadoras. No obstante, muchos problemas y limitaciones se manifestaron en este método, algunas de las desventajas de los sistemas que trabajan con ficheros son:

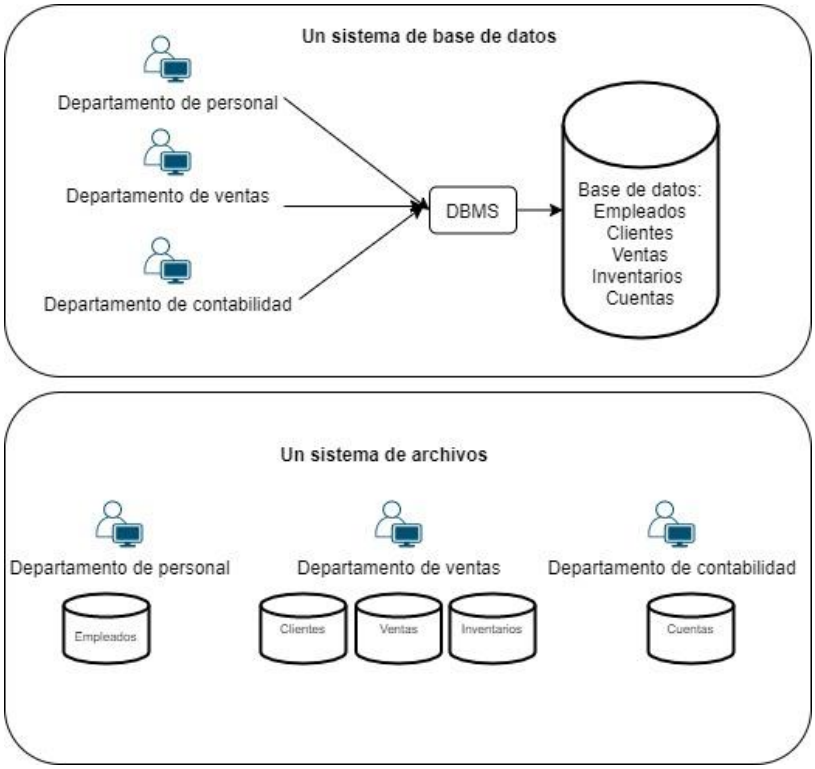
-La organización de los datos en los ficheros es totalmente dependiente de la aplicación que los trata, es decir, si es necesario cambiar la estructura de alguno de los ficheros, será también necesario modificar la aplicación que los usa.

- Si es necesario cambiar una aplicación, casi con total seguridad será necesario modificar el número de ficheros que usa, su organización, los campos, etc.
- Es casi seguro que existirá una alta redundancia de los datos, es decir, que existirán datos repetidos en diversos ficheros.

1.3.2 Sistemas de base datos

Los problemas propios de los sistemas de archivos hacen bastante deseable un sistema de bases de datos. A diferencia del sistema de archivos, con sus numerosos archivos separados y sin relación entre sí, el sistema de bases de datos está formado por datos relacionados lógicamente en un solo depósito lógico. Como el depósito de datos de la base de datos es una sola unidad lógica, la base de datos representa un cambio importante en la forma en que los datos del usuario final son guardados, se tiene acceso a ellos y son administrados.

Figura 7. Comparación de los sistemas de archivos y sistemas de base de datos (Coronel,2011).



1.4 Los distintos niveles de abstracción de una base de datos

Una base de datos con una buena arquitectura debe permitir su utilización en distintas máquinas con distintos sistemas operativos, es decir, admitir la

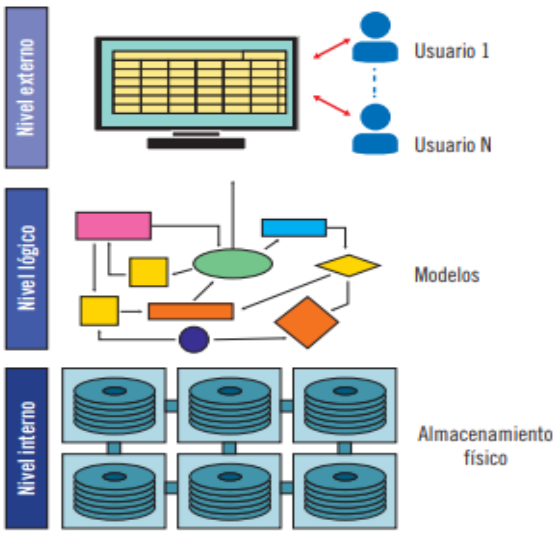
portabilidad. Otro de los objetivos es la abstracción de datos, lo que significa dar a los usuarios una visión abstracta de los datos, es decir, una visualización de los datos, pero no el conocimiento de la estructura interna.

La abstracción de datos consiste en proporcionar a los usuarios una visión abstracta de los datos, lo cual no implica que haya que mostrar ni conocer la estructura interna. Por lo tanto, el SGBD debe ocultar los detalles de almacenamiento y manejo ofreciendo estructuras de datos eficientes para un buen rendimiento.

Por ello, la arquitectura ANSI/SPARC permite ver una base de datos dividida en tres niveles de abstracción:

- Nivel físico (Interno): nivel más bajo y exhaustivo. Trata de los detalles del almacenamiento real.
- Nivel lógico: nivel que describe la información guardada y las relaciones de la información almacenada. La definición de estructuras de datos a este nivel puede suponer la creación de estructuras complejas a nivel físico.
- Nivel de visión (Externo): nivel de abstracción más alto, describe solo una parte de la base de datos puesto que es la que se muestra el usuario. También conocido como nivel conceptual.

Figura 8. Niveles de abstracción de una base de datos



1.5 Componentes del sistema gestor de bases de datos

Un sistema gestor de bases de datos (SGBD) es una colección de programas que facilitan la labor de gestionar la base de datos en su conjunto. En general, como indican Oltra et al. (2006) "el SGBD se encargará de gestionar el correcto funcionamiento interno de la base de datos, en lo que se refiere al control de la concurrencia y de la integridad, además de facilitar a los usuarios la creación, el mantenimiento y, en ocasiones, el diseño de dicha base de datos".

Los componentes principales de un SGBD son herramientas de gestión, herramientas de programación, lenguajes y el diccionario de datos.

Herramientas de gestión, todos los SGBD disponen de herramientas de gestión para poder crear las bases de datos, manipularlas, modificar su diseño, crear usuarios y asignar permisos, etc.

Herramientas de programación, estas herramientas posibilitan la creación de programas que accedan a los datos y los manipulen para su uso por parte de usuarios que no puedan o deban trabajar directamente con el SGBD.

Lenguajes, los SGBD proporcionan lenguajes que se pueden clasificar del siguiente modo:

- Lenguajes de definición de datos (LDD): estos lenguajes permiten la descripción de la estructura lógica global de la base de datos, de la estructura interna y de las estructuras externas que sean necesarias para el desarrollo de las diferentes aplicaciones.
- Lenguajes de manipulación de datos (LMD): estos lenguajes permiten a los usuarios efectuar consultas, inserciones, borrados y modificaciones sobre los datos de la base de datos.

El diccionario de datos, contiene toda la información sobre los datos almacenados en la base de datos. En una base de datos relacional, el diccionario de datos proporciona información acerca de:

- La estructura lógica y física de la base de datos.
- Las definiciones de todos los objetos de la base de datos: tablas, vistas, índices, disparadores, procedimientos, funciones, etc.
- El espacio asignado y utilizado por los objetos.

- Los valores por defecto de las columnas de las tablas.
- Información acerca de las restricciones de integridad.
- Los privilegios y roles otorgados al usuario.
- Auditoría de información, como los accesos a los objetos.

1.6 Bases de datos relacionales, basadas en objetos y semiestructuradas.

El modelo relacional fue introducido en la década de 1970 por E. F. Codd (de IBM) en su destacado artículo científico "A Relational Model of Data for Large Shared Databanks" (Communications of the ACM, junio de 1970, pp. 377-387). El modelo relacional representó un importante avance para usuarios y diseñadores. Para usar una analogía, el modelo relacional produjo una base de datos de "transmisión automática" para sustituir las bases de datos de "transmisión estándar" que le precedieron. Su sencillez conceptual preparó el terreno para una genuina revolución en las bases de datos.

Sin embargo, fue en 1979 la Compañía de Larry Ellison pudo vender la primera base de datos relacional basada en SQL, mucho antes que IBM. Para 1983, la compañía había producido una versión portátil de la base de datos que generó ingresos brutos anuales de más de $5 000 000 y cambió su nombre a Oracle. En 1980, estimulada por la competencia, IBM finalmente produjo SQL/DS, su primera base de datos relacional.

1.6.1 El modelo orientado a objetos

En el modelo de datos orientado a objetos (OODM), tanto los datos como sus relaciones están contenidos en una sola estructura conocida como objeto.

Un OODM refleja una forma muy diferente de definir y usar entidades. Al igual que la entidad de un modelo relacional, un objeto está descrito por su contenido fáctico. Pero, a diferencia de una entidad, un objeto incluye información acerca de las relaciones entre los datos dentro del objeto, así como información acerca de sus relaciones con otros objetos. Por tanto, a los datos dentro del objeto se les da un mayor significado.

El OODM es un modelo de datos semántico porque semántico indica significado. Es importante también conocer los tipos de relaciones que existen en el modelo orientado a objetos son relación de asociación, de generalización y de agregación.

1.6.2 Datos semiestructurados

Los datos semiestructurados son datos que ya han sido procesados en alguna medida. Por ejemplo, si observamos una página web cualquiera, los datos se presentan en formato arreglado previamente para presentar alguna información.

Las necesidades de almacenamiento y administración de datos no estructurados y semiestructurados se manejan por medio de una generación de bases de datos conocida como bases de datos XML y Json. El Lenguaje de marcado extensible (*Extensible Markup Language* XML), por sus siglas en inglés es un lenguaje especial que se emplea para representar y manipular elementos de datos en un formato textual. Una base de datos XML soporta el almacenamiento y la administración de datos XML semiestructurados.

1.6.1 Datos desestructurados o no estructurados

Son aquellos que no tienen ningún formato específico, su estructura normalmente no es uniforme y se tiene poco o nulo control sobre ellos, la información no está representada por datos elementales y su interpretación y manipulación es más compleja.

Los archivos que se pueden encontrar datos no estructurados son archivos PDF, correo electrónico, música, archivos JPG, videos, documento de texto. Hay dos tipos de datos no estructurados los generados por máquina y por los seres humanos.

Algunos de los ejemplos de datos no estructurados generados por máquinas son generados por satélite: datos del tiempo y meteorológicos; datos científicos: sísmicos, atmosféricos, físicos en general; fotografía y video: datos de seguridad, video, tráfico, de vigilancia; de radar o sonar: perfiles oceanográficos.

Ejemplos de datos no estructurados generados por seres humanos se dan en información interna de empresas: documentos, informes, emails, cualquier tipo de información textual que se intercambia diariamente en la empresa; de redes sociales: youtube, Facebook, twitter; datos de móviles: SMS, de geolocalización;

datos de contenido web: contenidos sin estructura generados por cualquier sitio web.

1.7 Arquitectura de los sistemas de bases de datos.

Los SBD (Sistemas de base de datos) necesitan una descripción o definición de la base de datos, esta descripción recibe el nombre de esquema de la base de datos, y los SBD la tendrán continuamente a su alcance.

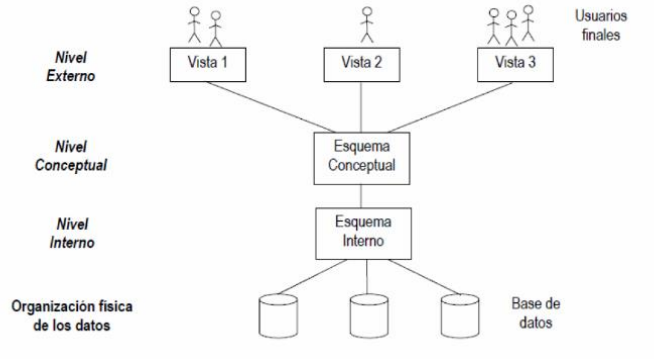
El esquema de la base de datos es un elemento fundamental de la arquitectura de un SBD que permite independizar el SBD de la base de datos; de este modo, se puede cambiar el diseño de la base (su esquema) sin tener que hacer ningún cambio en el SBD.

De acuerdo con la arquitectura ANSI/SPARC, en un sistema de base de datos debe haber tres niveles de esquemas (tres niveles de abstracción). La idea básica de ANSI/SPARC consistía en descomponer el nivel lógico en dos: el nivel externo y el nivel conceptual. Denominábamos nivel interno lo que aquí hemos denominado nivel físico.

De este modo, de acuerdo con ANSI/SPARC, habría los tres niveles de esquemas que mencionamos a continuación:

a) En el nivel externo se sitúan las diferentes visiones lógicas que los procesos usuarios (programas de aplicación y usuarios directos) tendrán de las partes de la base de datos que utilizarán. Estas visiones se denominan esquemas externos.
b) En el nivel conceptual hay una sola descripción lógica básica, única y global, que denominamos esquema conceptual, y que sirve de referencia para el resto de los esquemas.
c) En el nivel físico hay una sola descripción física, que denominamos esquema interno.

Figura 9. Arquitectura del sistema de base de datos



1.8 Cuestionario de la unidad

1. ¿Qué es la información?

Es el resultado de procesar datos sin elaborar para dejar ver su significado.
Es un grado de espontaneidad manifestado a través de una gráfica.
Son todos aquellos que se pueden visualizar en tablas.
Son los referentes a cálculos estadísticos.

2. ¿Cuál es el propósito final de los sistemas de información en todos los negocios?

Ayudarlos a usar la información como un recurso organizacional.
Ayudarlos a organizar el recurso humano eficientemente.
Ayudarlos a optimizar procesos de inventario.
Ayudarlos a evadir riesgos innecesarios.

3. ¿Cuál es el corazón de todos los sistemas de información

La captura, el almacenamiento, agregado, la manipulación, la diseminación y la administración de datos.
El recurso humano institucional, la satisfacción de la alta gerencia, la continuidad del negocio.
La forma de mantenerse vigente en el mercado y dar oportunidad de desarrollo a los departamentos de TIC.
El recurso que preste el departamento de tecnología institucional junto a las interpretaciones que sepa absorber de gerencia.

4. ¿A qué hace referencia el término “base de datos”?

A un conjunto de datos multidimensionales en el sentido de que los enlaces internos existentes entre los distintos elementos hacen que se pueda acceder a la información con diversas perspectivas.
A que hay un sistema de almacenamiento unidimensional, lo que quiere decir que presenta la información que contiene desde un único punto de vista.

A componentes que definen y regulan la captura, almacenamiento, administración y uso de datos dentro de un ambiente de base de datos.
A que existen mecanismos de defensa por posibles pérdidas de integridad en la data de la organización.

5. ¿A qué se refiere el término "sistema de base de datos"?

A componentes que definen y regulan la captura, almacenamiento, administración y uso de datos dentro de un ambiente de base de datos.
A un conjunto de datos multidimensionales en el sentido de que los enlaces internos existentes entre los distintos elementos hacen que se pueda acceder a la información con diversas perspectivas.
A que hay un sistema de almacenamiento unidimensional, lo que quiere decir que presenta la información que contiene desde un único punto de vista.

6. ¿Cuáles son las partes principales de un "sistema de base de datos"?

Hardware, software, personas, procedimientos y datos.
Componentes administrativas establecidos por el DBA.
Sistema operativo, software DBMS y programas de aplicación y utilerías.
Dato e información.

7. ¿Qué es una regla de negocios?

Es una política, procedimiento o norma adoptada por una organización.
Es un criterio conservado por el diseñador de base de datos en función de su análisis.
Son estándares nacionales referidos a las formas de tributar y funcionar de una empresa.
Son mecanismos de seguridad en integridad y consistencia de bases de datos.

8. ¿Cuál es el hilo conductor de las tecnologías de almacenes de datos, procesamiento analítico en línea (OLAP) y data mining?

Sistemas de bases de datos.

Bases de datos.

Información.

Sistemas de información.

9. ¿En qué archivos se pueden encontrar datos no estructurados?

Archivos pdf, correo electrónico, música, archivos JPG, videos, documento de texto.

Archivos LDF y MDF.

Archivos de configuración de un gestor de base de datos.

Instrucciones del lenguaje JAVA

10. ¿Quién fue y en qué año definió las bases de datos relacionales o modelo relacional?

Edgar Codd, 1970

Peter Pin-Shan Chen, 1976

Peter Pin-Shan Chen a finales de los años 60

2 DISEÑO DE BASE DE DATOS

Este capítulo está dedicado a aprender a modelar y representar gráficamente una base de datos, a detectar los posibles problemas de diseño antes de que estos afecten a la aplicación, y a construir bases de datos óptimas para los distintos casos de relaciones entre entidades que formarán nuestra base de datos.

Objetivos:

- Conocer la estructura de una base de datos
- Desarrollar modelos Entidad-Relación
- Identificar cuál es la función del modelo de datos.

2.1 Estructuras de las bases de datos relacionales.

Para diseñar un base de datos relacional el primer paso es el modelo de datos, el modelo de datos es un conjunto de herramientas conceptuales que permiten describir los datos, sus relaciones, límites de integridad que les afectan, así como la terminología a emplear. Algunos de los principales modelos son *Abrial, Entity*

Relationship, Extended ER, RM/T, UML, los elementos básicos de todos los modelos de datos son entidades, atributos, relaciones y restricciones

Una de las características que al menos debe contener un modelo de datos a implementar es tener la Descripción de estructura de datos que guardará, conjunto de reglas que garantizan la integridad de datos, metodología de manipulación de datos para apoyar las transformaciones de los datos reales.

Estructuras de datos y sus características, relaciones, restricciones, transformaciones y otras construcciones con el propósito de sostener un dominio del problema específico es todo lo que representa el modelo de base de datos dentro del ambiente de base de datos. La evolución de los modelos de datos a través del tiempo ha iniciado por sistemas de archivos, jerárquicos y de red, relacional, orientado a objetos hasta DBMS híbrido en XML**.**

El principal objetivo del modelo de datos relacional es facilitar que la base de datos sea percibida o vista por el usuario como una estructura lógica que consiste en un conjunto de relaciones y no como una estructura física de implementación. Con este objetivo se logra conseguir un alto grado de independencia de los datos.

2.2 Operaciones fundamentales del álgebra relacional.

El álgebra relacional origina en la teoría de conjuntos. El álgebra relacional consta de un conjunto de operadores que toman una o más relaciones como operandos y producen otra relación como resultado. Se compone de dos grupos de operadores: operadores tradicionales de la teoría de conjuntos y operadores relacionales específicos.

El estudio del álgebra relacional presenta un interés especial, pues ayuda a entender qué servicios de consulta debe proporcionar un lenguaje relacional, facilita la comprensión de algunas de las construcciones del lenguaje SQL y también sirve de base para el tratamiento de las consultas que efectúan los SGBD internamente.

Una característica destacable de todas las operaciones del álgebra relacional es que tanto los operandos como el resultado son relaciones, a esta propiedad se la denomina cierre relacional.

Las operaciones del álgebra relacional han sido clasificadas según distintos autores; de todos ellos se podría indicar 3 operaciones:

1) Según se pueden expresar o no en términos de otras operaciones.

a) Operaciones primitivas: son aquellas operaciones a partir de las cuales podemos definir el resto. En estas operaciones encontramos la unión, la diferencia, el producto cartesiano, la selección y la proyección.

b) Operaciones no primitivas: el resto de las operaciones del álgebra relacional que no son estrictamente necesarias, porque se pueden expresar en términos de las primitivas; sin embargo, las operaciones no primitivas permiten formular algunas consultas de forma más cómoda.

2) Según el número de relaciones que tienen como operandos:

a) Operaciones binarias: son las que tienen dos relaciones como operandos. Son binarias todas las operaciones, excepto la selección y la proyección.

b) Operaciones unarias: son las que tienen una sola relación como operando. La selección y la proyección son unarias.

3) Según se parecen o no a las operaciones de la teoría de conjuntos:

a) Operaciones conjuntistas: son las que se parecen a las de la teoría de conjuntos. Se trata de la unión, la intersección, la diferencia y el producto cartesiano.

b) Operaciones específicamente relacionales: son el resto de las operaciones; es decir, la selección, la proyección y la combinación.

2.2.1 Notaciones para el álgebra relacional

A continuación, se mostrará cuáles son los operadores para notaciones formales del algebra relacional.

Tabla 1 Notaciones Algebra Racional

Operador	Sintaxis	Definición
Restrict	Transforma una única relación R en las tuplas resultantes que casan con la condición específica (predicado).	**σ predicado (R)**
Project	Transforma una única relación R en un subconjunto formado por los atributos especificadores σσσσσσ.	**Π a_1……..a^n (R)**
Union	Produce un resultado a partir de dos relaciones R Y S ,	**R ∪ S**

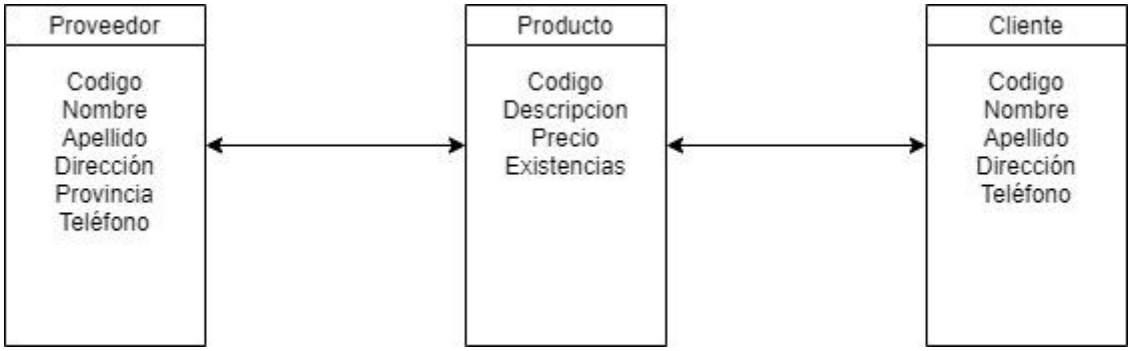
	están contienen las tuplas R o S, o tanto de R como de S.	
Difference	Produce un resultado a partir de dos relaciones R y S en el cual las tuplas devueltas existen en R pero no en S.	**R - S**
Intersection	Produce un resultado a partir de dos relaciones R y S	**R S**
Natural join	Un natural join es una equi-join de dos relaciones R y S.	**R [> <] S**

2.3 El Modelo entidad – relación.

2.3.1 Modelo Relacional

Un diagrama relacional es una representación de las entidades de la base de datos relacional, los atributos dentro de esas entidades y las relaciones entre esas entidades. Los aspectos fundamentales del modelo de datos relacional son estructura de datos, integridad de datos, manipulación de datos.

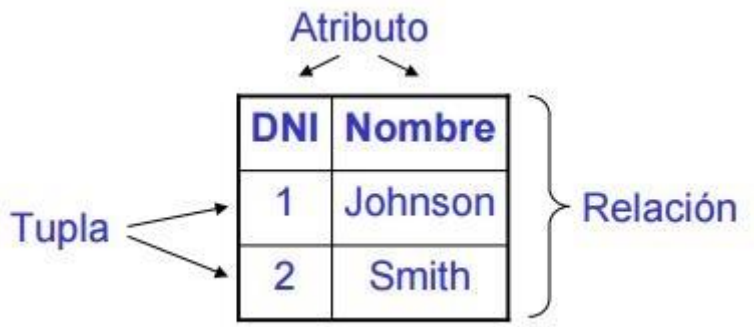
Figura 10. Modelo Relacional



El modelo relacional de datos se implementa por medio de un complejo sistema de administración de base de datos relacional (RDBMS, por sus siglas en inglés). El RDBMS realiza las mismas funciones básicas de los sistemas DBMS jerárquico y de red, además de una amplia variedad de otras funciones que hacen que el modelo de datos relacional sea más fácil de entender e implementar.

Los datos en los modelos relacionales están almacenados representativamente en rectángulos, denominados relaciones. Una fila de una relación se denomina tupla, las columnas de una relación se denominan atributos porque cada entrada de una columna describe alguna característica o atributo de la entidad representada por la correspondiente tupla.

Figura 11. Tupla-Atributos



2.3.2 Modelo entidad – relación (ER)

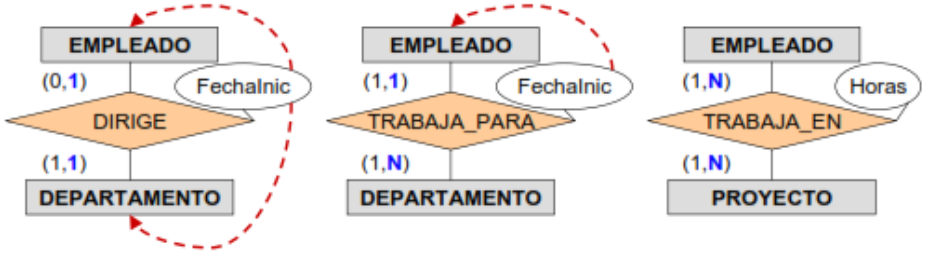
Peter Chen introdujo inicialmente el modelo ER en 1976; fue la representación gráfica de entidades y sus relaciones en una estructura de base de datos que con rapidez se hizo popular porque complementaba los conceptos del modelo de datos relacional. El modelo de datos relacional y el ERM se combinaron para fundamentar el diseño de base de datos estrechamente estructurado.

Es el modelo conceptual más utilizado, basado en una percepción del mundo real mediante una colección de objetos, que se denominan entidades y relaciones. El modelado semántico interpreta en el modelo entidad relación por medio de los datos que se encuentran en un sistema de información habitualmente no caen irrevocablemente en las categorías entidad, relación, o atributo.

El modelo ER está basado en los siguientes componentes:

- Entidad. Una entidad se representa en el ERD con un rectángulo, también conocido como caja de entidad. El nombre de la entidad, un sustantivo, se escribe en el centro del rectángulo. El nombre de entidad por lo general se escribe en mayúsculas y en forma singular: *PAINTER , EMPLOYEE*
- Relaciones. Las relaciones describen asociaciones entre datos. Casi todas las relaciones describen asociaciones entre dos entidades. Cuando se explicaron los componentes básicos del modelo de datos se ilustraron tres tipos de relaciones entre datos: uno a muchos (1:M), muchos a muchos (M:N) y uno a uno (1:1). El modelo ER utiliza el término conectividad para etiquetar los tipos de relación. El nombre de la relación suele ser un verbo activo o pasivo. Por ejemplo, un EMPLEADO dirige un DEPARTAMENTO; un EMPLEADO trabaja en muchos PROYECTOS; un EMPLEADO trabaja en muchos DEPARTAMENTOS.

Figura 12. Ejemplo modelo



ER

2.4 Diagramas entidad - relación.

Los diagramas entidad-relación es una herramienta para el modelado de datos, el resultado final al realizar un diagrama ER es un modelo gráfico de las entidades y las relaciones en un particular domino del discurso.

Las notaciones típicas con las que se pueden trazar un diagrama entidad-relación son Chen tradicional, pata de gallo, diagrama de clase UML.

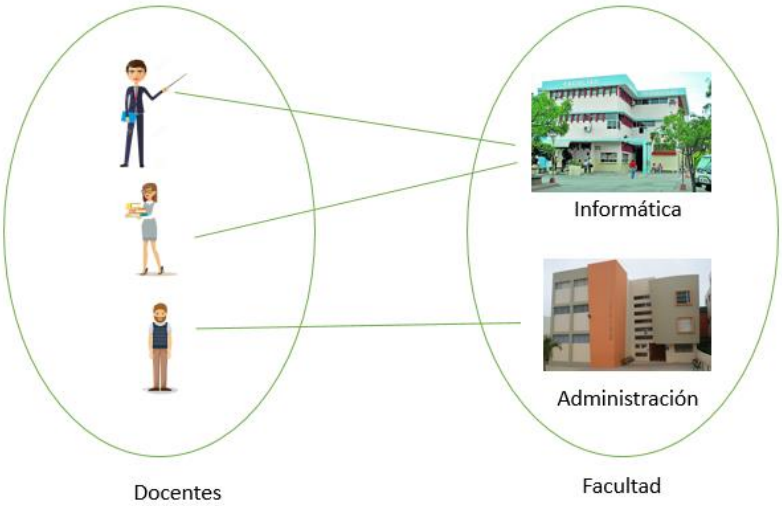
Los diseñadores de bases de datos por lo general usan las notaciones breves 1:M o 1..* M:N o *..* y 1:1 o 1..1, respectivamente. (Aun cuando la notación M:N es una leyenda estándar para la relación de muchos a muchos, la notación M:M también puede usarse.) Los siguientes ejemplos **ilustran las distinciones entre las tres.**

2.4.1 Diagrama de instancia

Los diagramas de instancia son utilizados para comprender de una forma más fácil los conceptos de Cardinalidad y participación. Estos diagramas muestran cuantas ocurrencias o instancias de las entidades relaciona, por ello cada entidad se representa como un conjunto de instancias.

Un diagrama de instancia en el modelo entidad relación son representados con óvalos que se utilizan para representar conjuntos de instancias que indican cuantas ocurrencias de las entidades se relacionan.

Figura 13. Una relación uno a muchos como diagrama de instancia



2.5 Nivel conceptual de los datos.

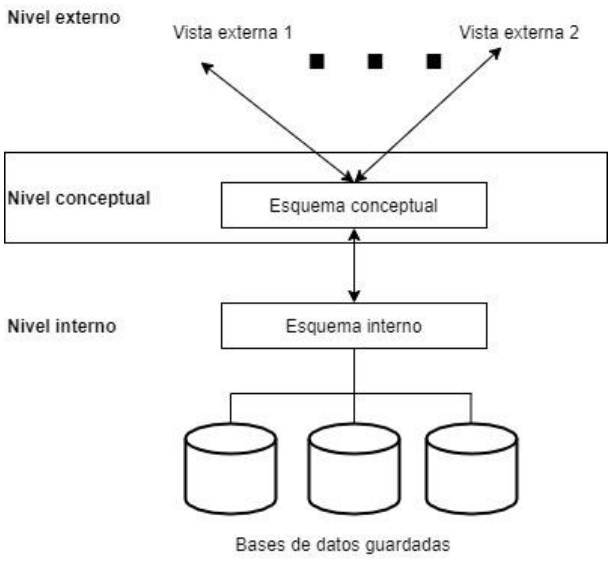
Las necesidades de modelar datos son una función de diferentes vistas de datos (global vs. local) y el nivel de abstracción de datos. El American National Standards Institute, Standards Planning and Requirements Committee (ANSI/SPARC) describe tres niveles de abstracción de datos: externos, conceptuales e internos.

En el nivel conceptual se describe la estructura de toda la base de datos para una comunidad de usuarios (todos los de una empresa u organización), mediante un esquema conceptual. Este esquema oculta los detalles de las estructuras de almacenamiento y se concentra en describir entidades, atributos, relaciones, operaciones de los usuarios y restricciones. En este nivel se puede utilizar un modelo conceptual o un modelo lógico para especificar el esquema.

Este nivel no especifica cómo se almacenan físicamente los datos, los puntos más importantes acerca de este nivel son:

1. El DBA (Administrador de la base de datos) trabaja en este nivel.
2. Describe la estructura de todos los usuarios.
3. Solo el DBA puede definir este nivel.
4. Visión global de la base de datos.
5. Independiente de hardware y software.

Figura 14. Nivel conceptual



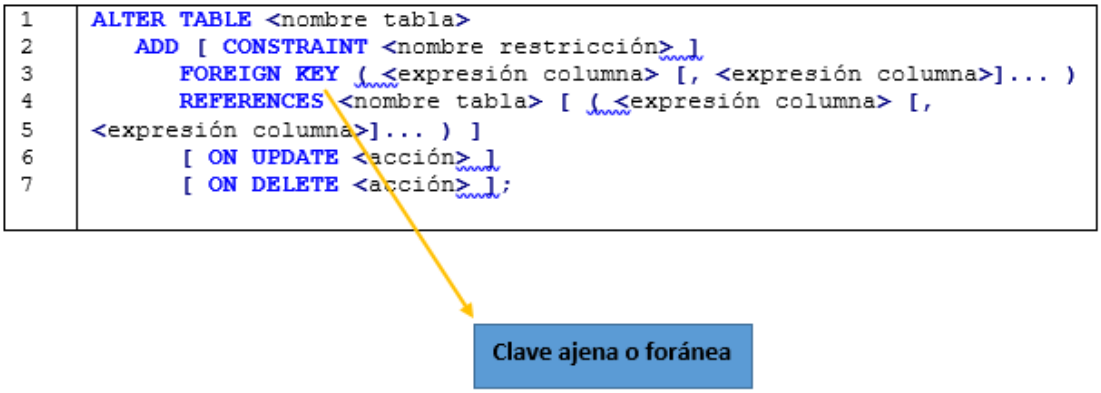
2.6 Conectividad y Cardinalidad: Dependencia de existencia, entidades débiles

2.6.1 Claves ajenas

Una clave ajena o foránea es una columna o los grupos de columnas de una tabla que toma sus valores del mismo dominio que la clave primaria de una tabla de la misma base de datos con la que está relacionada. Los valores que puede tomar la clave foránea son todos que haga referencia a un valor de clave primaria de alguna de las tablas de la base de datos, o, nulo.

Las claves foráneas permiten establecer conexiones entre las tuplas de las relaciones, para hacer posible la conexión, una clave foránea tiene el conjunto de atributos de una relación que referencian la clave primaria de otra relación (o incluso de la misma relación). Su principal objetivo es establecer una conexión con la clave primaria que referencian. Por lo tanto, los valores de una clave foránea deben estar presentes en la clave primaria correspondiente, o bien deben ser valores nulos. En caso contrario, la clave foránea representaría una referencia o conexión incorrecta.

Figura 15. Definiendo clave foránea



2.6.2 Entidades débiles

La entidad débil es aquella cuya existencia sí depende de que exista otra entidad en el universo del discurso, una entidad débil es la que satisface dos condiciones:

1. La entidad es dependiente de existencia: esto es, no puede existir sin la entidad con la que tiene una relación.

2. La entidad tiene una llave primaria que se deriva parcial o totalmente de la entidad padre en la relación.

2.7 Desarrollo de un diagrama ER

Para el desarrollo de un diagrama ER, se emplea tres conceptos básicos: conjuntos de entidades, conjuntos de relaciones y atributos.

Conjunto de entidades

Una entidad es una cosa u objeto del mundo real que es diferente de los demás objetos o cosas. Una entidad posee un conjunto de propiedades y los valores de estas propiedades identifican y distinguen a cada entidad de las otras. Hay dos tipos de entidades, las concretas y las abstractas.

Conjunto de relaciones

Una relación es una asociación entre dos o varias entidades, es decir es el vínculo que existe entre dos o más entidades. Un conjunto de relaciones es un conjunto de relaciones del mismo tipo. La asociación entre conjunto de entidades se conoce como participación. La función que desempeña una entidad en una relación se denomina rol de esa entidad.

Atributos

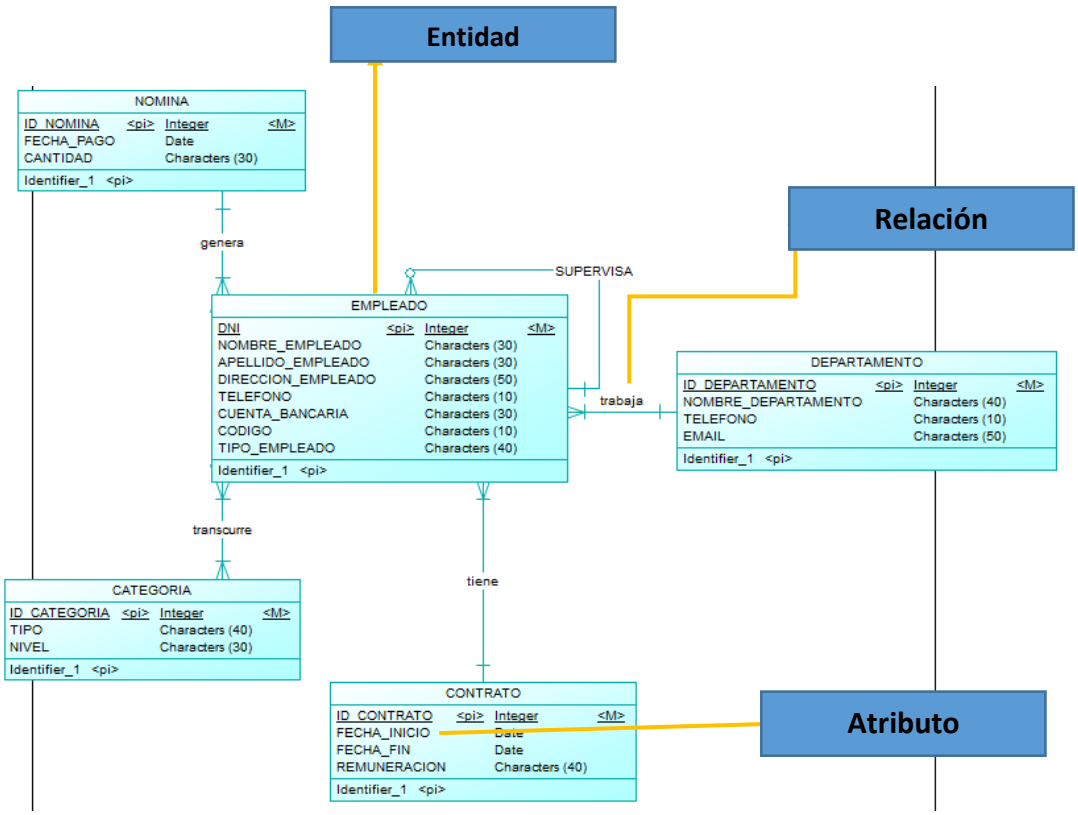
Los atributos en un diagrama E-R tienen un conjunto de valores permitidos que son conocidos con el nombre de dominio o conjunto de valores para el atributo.

Hay diferentes tipos de atributos:

- **Atributos simples y compuestos.** Los atributos simples son aquellos que no están divididos en subpartes. Los atributos compuestos si se dividen en subpartes (es decir, en otros atributos), por ejemplo, un atributo llamado nombre, puede estar formado por nombre, apellido_paterno, apellido_materno. O el atributo Dirección se puede descomponer en calle, número, ciudad, código_postal. Los atributos compuestos ayudan a agrupar atributos relacionados, lo que hace que los modelos sean más claros.
-

- **Atributos monovalorados y multivalorados.** Los atributos que poseen solo un valor se conocen con el nombre de atributos monovalorados, pero puede darse el caso que un atributo tenga un conjunto de valores para una entidad concreta, un atributo multivalorado es aquel que puede contener varios valores, como por ejemplo al atributo numero_telefono. En ocasiones es necesario establecer límites inferior y superior al número de valores de un atributo multivalorado.
- **Atributos derivados.** El valor de este tipo de atributos se puede obtener a partir del valor de otros atributos o entidades relacionadas. Por ejemplo supóngase el atributo edad, que se va a calcular a partir de otro atributo llamado fecha_nacimiento. Edad sería un atributo derivado y fecha de nacimiento un atributo básico o almacenado. El valor de los atributos derivados no se almacena, es decir no se inserta, se hacen cálculos para obtener su valor cada que sea necesario.

Figura 16. Modelo Relacional de datos



2.8 Paso del modelo E/R al modelo relacional

El modelo ER es un modelo de datos conceptual de alto nivel, este facilita las tareas de diseño conceptual de bases de datos, pero es necesario traducirlo a un esquema que sea compatible con un SGBD y el Modelo Relacional es utilizado por la mayoría de los SGBD existentes en el mercado.

Para pasar el diagrama entidad relación al modelo relacional, para cada entidad del diagrama E/R obtenemos una tabla, el atributo identificador de la entidad se convierte en la clave primaria de la tabla. El resto de los atributos de una entidad se convierten en atributos no clave de la tabla, para cada relación uno-a-muchos se pone la clave primaria de la tabla "uno" en la tabla que representa la parte "muchos" de la relación.

2.9 Normalización de Bases de Datos

La normalización es la forma de diseñar lógicamente una base de datos en un modelo relacional y consiste en la descomposición de los datos en varias tablas diferentes.

Los motivos por los que hay que normalizar una base de datos son:

- Evitar la redundancia de datos.
- Proteger la integridad de los datos.
- Evitar problemas de actualización de los datos en las tablas.

Para conseguir los objetivos de los tres puntos anteriores existen las formas normales, de las que a continuación se explican las tres primeras.

2.9.1 Primera forma normal (1FN)

Una relación está en primera forma normal si y solo si la intersección de cada una de sus filas y columnas contiene un valor único o valor atómico.

Una tabla no puede tener distintos valores en cada columna, ya que los datos no son atómicos, esto quiere decir que si a un valor **X** le pertenece un valor **Y**, a un valor **Y** le pertenece el valor **X**. Para ilustrar la 1FN, cree la típica tabla de un almacén que suministra mercancías a proveedores.

Ejemplo.

Los registros quedan ahora conformados en dos tablas que llamaremos ORDENES y ARTICULOS_ORDENES ordenes (id_orden, fecha, id_cliente, nom_cliente, estado)

Ordenes

Tabla 2. Ordenes

Id_orden	Fecha	Id_cliente	Nom_cliente	Estado
2301	23/02/11	101	Martin	Caracas
2302	25/02/11	107	Herman	Coro
2303	27/02/11	110	Pedro	Macaray

2.9.2 Segunda forma normal

Una relación se encuentra en segunda forma normal si y solo si está en primera forma normal y todos los atributos no clave dependen de la clave primaria.

De este modo, al añadir nuevos registros no se repetirán innecesariamente los mismos valores de un registro a otro.

Ejemplo:

Las tablas quedan ahora de la siguiente manera.

Articulos_ordenes (id_orden, num_art, cant)

Tabla 3. Articulos_ordenes

Id_orden	Num_art	Cant
2301	3786	3
2301	4011	6
2301	9132	8
2302	5794	4
2303	4011	2
2303	3141	2

2.9.3 Tercera forma normal

Una relación se encuentra en tercera forma normal si y solo si está en segunda forma normal y si, además, cada atributo no clave:

Es mutuamente independiente, es decir, no existe ningún atributo no clave que dependa funcionalmente de alguna combinación del resto de atributos no clave.

Cada atributo no clave es completamente dependiente de la clave primaria.

Básicamente, consiste en eliminar todos aquellos campos que pueden obtenerse a partir de campos existentes en la tabla que estamos normalizando o en otras tablas de la base de datos.

Ejemplo:

ordenes (id_orden, fecha, id_cliente)

Ordenes

Tabla 4. Tabla en tercera forma normal

Id_orden	Fecha	Id_cliente
2301	23/02/11	101
2302	25/02/11	107
2303	27/02/11	110

2.10 Cuestionario de la unidad

1. ¿Qué es un modelo de datos?

Es un conjunto de herramientas conceptuales que permiten describir los datos, sus relaciones, límites de integridad que les afectan, así como la terminología a emplear.

Son las formas de organizar la información en una organización.

Es un conjunto de tablas cuya estructura representa las unidades particulares de una organización.

Es aquel que está fundamentado en mentefactos del programador.

2. En orden respectivo ¿cuál es la evolución de los modelos de datos?

Sistemas de archivos, jerárquicos y de red, relacional, orientado a objetos, DBMS híbrido en XML.

Sistemas de archivos, jerárquicos y de red, orientado a objetos, relacional, DBMS híbrido en XML.

Jerárquicos y de red, sistemas de archivos, relacional, orientado a objetos, DBMS híbrido en XML.

Híbridos, normales, redundantes y de cohesión.

3. ¿Cuáles son los aspectos fundamentales del modelo de datos relacional?

Estructura de datos, integridad de datos, manipulación de datos.

Dominios de atributos y los valores que pueden asumir.

Estructuras de datos, sean estas lineales o jerárquicas.

Dominios de valor, dominios de seguridad e integridad.

4. ¿Cómo se implementa el modelo de datos relacional?

Por medio de un sistema de administración de base de datos relacional.

Por medio de mecanismos que acepten los conceptos entidad-relación.

Por medio de sistemas de aplicación correctamente configurados.

Por medio de redes y almacenamiento.

5. ¿Quién fue y en qué año creó el modelo entidad-relación?

Peter Pin-Shan Chen, 1976

Peter Pin-Shan Chen a finales de los años 80

Edgar Codd, 1976

Edgar Codd a finales de los años 60

6. ¿Para qué sirven las claves ajenas?

Son aquellas que interconectan los datos almacenados en tablas diferentes.

Es una columna o un grupo de columnas que pueden actuar como identificador único.

Es una o más columnas de una tabla cuyos valores se usan para identificar de forma única cada una de las filas de la tabla.

Es un valor único en toda la base de datos.

7. ¿Qué es una entidad débil en el modelo entidad relación?

Aquella cuya existencia sí depende de que exista otra entidad en el universo del discurso.

Aquella que no depende de la existencia de otra entidad y tiene al menos una relación.

Aquella cuya existencia sí depende de que exista otra entidad en el universo del discurso, y tiene varias relaciones.

8. ¿Qué es una base de datos relacional?

Es aquella que representa los datos y las relaciones entre los datos mediante una colección de tablas.

Son estructuras de datos que pertenecen al modelo jerárquico.

Es aquella que representa los datos mediante estructuras de datos de red.

Son estructuras de datos que distinguen entidades y atributos de esas entidades.

9. ¿Cómo se pasa el diagrama entidad relación al modelo relacional?

Para cada entidad del diagrama E/R obtenemos una tabla, el atributo identificador de la entidad se convierte en la clave primaria de la tabla. El resto de los atributos de una entidad se convierten en atributos no clave de la tabla, para cada relación uno-a-muchos se pone la clave primaria de la tabla "uno" en la tabla que representa la parte "muchos" de la relación.

Para cada entidad del diagrama E/R obtenemos una clave primaria. El atributo identificador de la entidad se convierte debe buscar claves ajenas, se dejan listadas las claves candidatas, al resto de los atributos de una entidad se les da dominio.

Para cada entidad del diagrama E/R obtenemos claves primarias y foráneas, el atributo identificador de la entidad se evalúa con el resto de los atributos de una entidad, el resto de los atributos se ubican normalmente en tablas.

10. ¿Cuáles son los motivos para normalizar una base de datos

Evitar la redundancia de datos, proteger la integridad de los datos, evitar problemas de actualización de los datos en las tablas.

Obtener información de calidad que sirva para la toma de decisiones de la empresa.

Verificar que en los niveles de abstracción se definieron tipos de datos adecuados.

Obtener un diccionario de datos que guíe los procesos del administrador de bases de datos.

3. Lenguaje de Consulta Estructurado SQL

En el presente capítulo se identifica clausulas SQL para la definición de condiciones de búsqueda, agrupación y orden en la recuperación de datos dentro del modelo relacional. Para ello se dará a conocer el lenguaje de consulta estructurado SQL, como afecta sus funcionalidades y procedimientos.

Objetivos

- Consultar, actualizar y reorganizar datos.
- Crear y modificar la estructura de un sistema de base de datos.
- Controlar el acceso a sus datos.

3.1 Lenguajes de bases de datos

Un lenguaje de base de datos permite crear estructuras de bases de datos y tablas para realizar tareas básicas de administración de datos (agregar, eliminar y modificar) y efectuar consultas complejas diseñadas para transformar datos sin elaborar información útil. Además, un lenguaje de bases de datos debe realizar tales funciones básicas con mínimo esfuerzo de parte del usuario y su estructura y sintaxis de comandos debe ser fácil de aprender.

Dentro de lo que implica el análisis de este capítulo, se tiene en cuenta que un Sistema Manejador de Base de Datos (DBMS) proporciona acceso a los datos por medio de un lenguaje de consulta. Un lenguaje de consulta no es un lenguaje de procedimiento; es un lenguaje que permite al usuario especificar qué debe hacerse sin tener que especificar cómo debe hacerse.

3.1.1 Modelo relacional.

El modelo relacional representó un importante avance para usuarios y diseñadores. Para usar una analogía, el modelo relacional produjo una base de datos de "transmisión automática" para sustituir las bases de datos de "transmisión estándar" que le precedieron.

Cualquier aplicación de base de datos relacional basada en SQL comprende tres partes: una interfaz de usuario, un conjunto de tablas guardadas en la base de datos y el "motor" de SQL. Cada una de estas partes se explica a continuación.

- *La interfaz de usuario final*. Básicamente, la interfaz permite al usuario final interactuar con los datos. Cada interfaz es un producto de la idea del vendedor de software de la interacción significativa con los datos. El usuario también puede diseñar su propia interfaz personalizada, con ayuda de generadores de aplicación que ahora son pan de todos los días en el campo del software de bases de datos.
- *Un conjunto de tablas guardadas en la base de datos*. En una base de datos relacional, todos los datos se perciben para estar guardados en tablas. Las tablas simplemente "presentan" los datos al usuario final en forma que sea fácil de entender. Cada tabla es independiente. Filas en diferentes tablas están relacionadas por valores comunes en atributos comunes.
- *Motor de SQL*. Casi por entero oculto al usuario final, el motor de SQL ejecuta todas las consultas, o peticiones de datos. Recuerde que el motor de SQL es parte del software del DMBS. El usuario final usa SQL para crear estructuras de tablas y ejecutar acceso a datos y mantenimiento de tablas. El motor de SQL procesa todas las peticiones del usuario, principalmente tras el escenario y sin que lo sepa el usuario final. En consecuencia, se dice que SQL es un lenguaje declarativo que dice lo que debe hacer, pero no cómo hacerse.

3.1.1.1 REDATAM

Es una herramienta utiliza una base de datos comprimida con millones de registros de personas, viviendas, manzanas de ciudades o cualquier división administrativa de un país.

El contexto más típico para su uso es el Censo de Población y Vivienda.

Figura 17. Investigación estadística variable matrimonio sistema REDATAM

Recuperado de:
http://redatam.inec.gob.ec/cgibin/RpWebEngine.exe/PortalAction?&MODE=MAIN&BASE=VITAL2012&MAIN=WebServerMain.inl

3.1.1.2 Caso de estudio Redatam

Para el caso de estudio a continuación se tomará la muestra de la variable inicial llamada "matrimonio", esta va a contar con variables subsiguientes como mes, número de hijos, edad, matrimonios anteriores, nivel de instrucción, provincia, entre otros.

Figura 18. Variable matrimonio sistema REDATAM

http://redatam.inec.gob.ec/cgibin/RpWebEngine.exe/PortalAction?&MODE=MAIN&BASE=VITAL2012&MAIN=WebServerMain.inl

3.1.1.3 Realización del caso de estudio Power Designer

PowerDesigner DataArchitect es la herramienta lidera de modelización de datos. Permite fortalecer y alinear negocio y IT. Power Designer permite a las empresas visualizar, analizar y manipular de manera más fácil los metadatos para tener una arquitectura de información de empresa eficaz.

Para realizar el caso de estudio en esta herramienta se tomará en cuenta las siguientes acciones.

1. **Realización de tablas.**

Para este ejemplo se elaborarán cinco tablas maestras cuyos nombres serán hijo, matrimonio, persona, provincia y testigo.

Tabla hijo

Tabla 5. Sentencia SQL de creación de la tabla hijo

```
1  create table HIJO (
2     IDHIJO                 INT4                 not null,
3     IDMATRIMONIO           INT4                 not null,
4     CANTIDADH              INT4                 null,
5     constraint PK_HIJO primary key (IDHIJO)
6  );
```

Figura 19. Tabla hijo en notación Chen tradicional

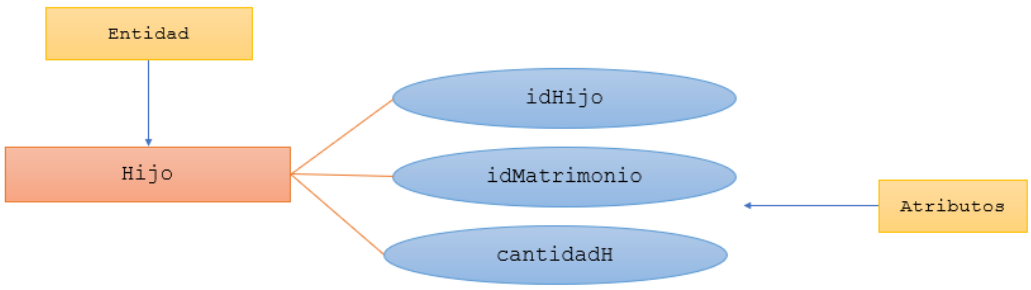


Tabla matrimonio

Tabla 6. Sentencia SQL de creación de la tabla matrimonio

```
create table MATRIMONIO (
   IDMATRIMONIO          INT4                not null,
   IDPERSONA             INT4                not null,
   IDPROVINCIA           INT4                not null,
   PROVINCIAM            CHAR(50)            null,
   FECHAM                DATE                null,
   CANTONM               CHAR(50)            null,
   constraint PK_MATRIMONIO primary key (IDMATRIMONIO)
);
```

Figura 20. Tabla matrimonio en notación Chen tradicional

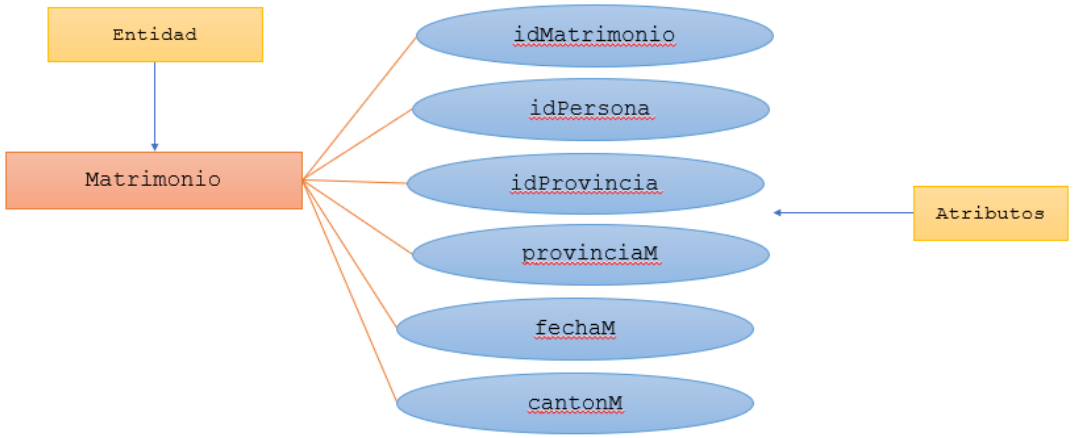


Tabla persona

Tabla 7. Sentencia SQL de creación de la tabla persona

```
create table PERSONA (
   IDPERSONA             INT4                not null,
   NOMBREP               CHAR(50)            null,
   CEDULAP               NUMERIC(10)         null,
   GENEROP               CHAR(10)            null,
   EDADP                 NUMERIC(999)        null,
   constraint PK_PERSONA primary key (IDPERSONA)
);
```

Figura 21. Tabla persona en notación Chen tradicional

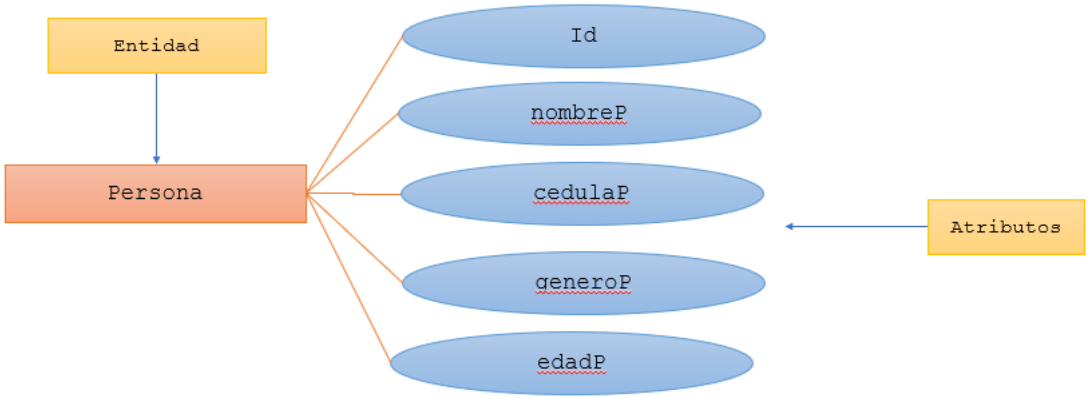


Tabla Provincia

Tabla 8. Sentencia SQL de creación de la tabla provincia

```
1  create table PROVINCIA (
2     IDPROVINCIA           INT4                  not null,
3     NOMBREPROVINCIA       CHAR(50)              null,
4     REGIONPROVINCIA       CHAR(50)              null,
5     constraint PK_PROVINCIA primary key (IDPROVINCIA)
6  );
```

Figura 22. Tabla provincia en notación Chen tradicional

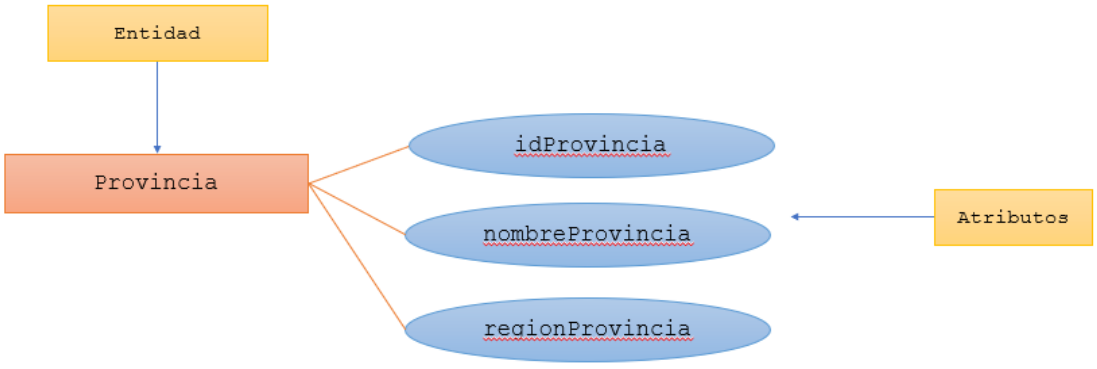
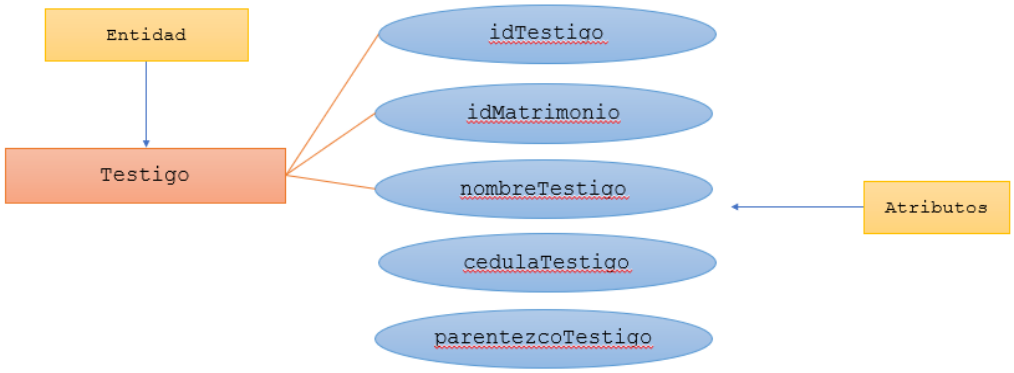


Tabla Testigo

Tabla 9. Sentencia SQL de creación de la tabla testigo

```
1  create table TESTIGO (
2     IDTESTIGO             INT4                  not null,
3     IDMATRIMONIO          INT4                  null,
4     NOMBRETESTIGO         CHAR(50)              null,
5     CEDULATESTIGO         NUMERIC(10)           null,
6     PARENTEZCOTESTIGO     CHAR(50)              null,
7     constraint PK_TESTIGO primary key (IDTESTIGO)
8  );
```

Figura 23. Tabla testigo en notación Chen tradicional



De estas tablas maestras va a relacionarse una con otra de la siguiente forma:

Tabla hijo – tabla matrimonio

Esta relación tiene una cardinalidad de cero a muchos (0, n), ya que en un matrimonio puede tener 0 hijos, un hijo o muchos hijos.

Tabla 10. Sentencia SQL de la tabla resultante entre hijo y matrimonio

```
1  alter table HIJO
2     add constraint FK_HIJO_RELATIONS_MATRIMON foreign
3  key (IDMATRIMONIO)
4        references MATRIMONIO (IDMATRIMONIO)
5        on delete restrict on update restrict;
```

Figura 24. Tabla de relación formada entre las entidades hijo y matrimonio en notación Chen tradicional

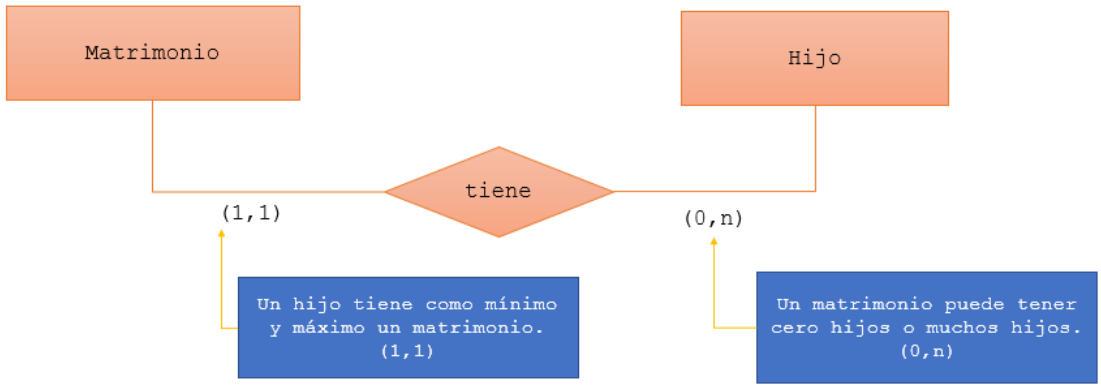


Tabla matrimonio – tabla persona

Esta relación tiene una cardinalidad de cero a muchos (0, n), ya que una persona puede o no puede casarse y a su vez una persona puede contraer un matrimonio o haberse casado varias veces.

Tabla 11. Sentencia SQL de la tabla matrimonio- tabla persona

```
1  alter table MATRIMONIO
2     add     constraint     FK_MATRIMON_RELATIONS_PERSONA
3  foreign key (IDPERSONA)
4        references PERSONA (IDPERSONA)
5        on delete restrict on update restrict;
```

Figura 25. Tabla matrimonio-persona en notación Chen tradicional

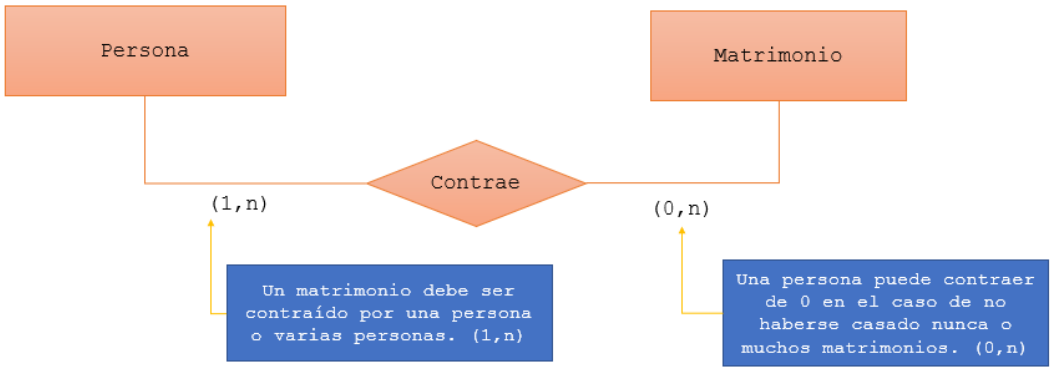


Tabla matrimonio – tabla provincia

Esta relación tiene una cardinalidad de uno a uno (1, 1), ya que un matrimonio se realiza en una sola provincia.

Tabla 12. Tabla matrimonio-Tabla provincia

```
1  alter table MATRIMONIO
2     add     constraint     FK_MATRIMON_RELATIONS_PROVINCI
3  foreign key (IDPROVINCIA)
4        references PROVINCIA (IDPROVINCIA)
5        on delete restrict on update restrict;
```

Figura 26. Tabla matrimonio-provincia en notación Chen tradicional

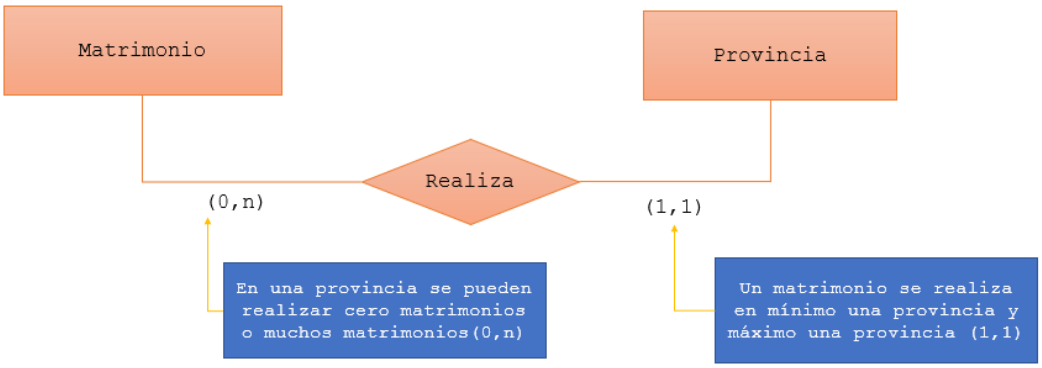


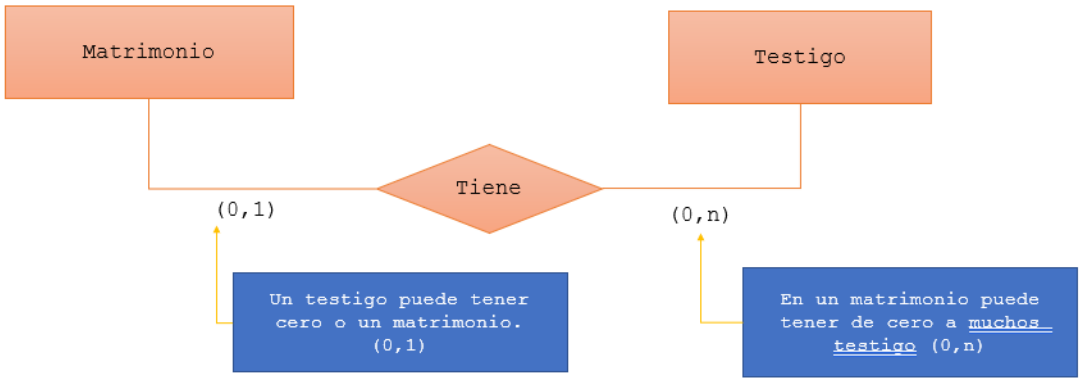
Table testigo -tabla matrimonio.

Esta relación tiene una cardinalidad de cero a muchos (0, n), en vista de que no es necesario tener un testigo para poder casarse, y a su vez se puede tener más de un testigo.

Tabla 13. Sentencia SQL de creación de la tabla testigo-tabla matrimonio

```
1  alter table TESTIGO
2     add     constraint     FK_TESTIGO_RELATIONS_MATRIMON
3  foreign key (IDMATRIMONIO)
4        references MATRIMONIO (IDMATRIMONIO)
5        on delete restrict on update restrict;
```

Figura 27. Tabla testigo-matrimonio en notación Chen Tradicional

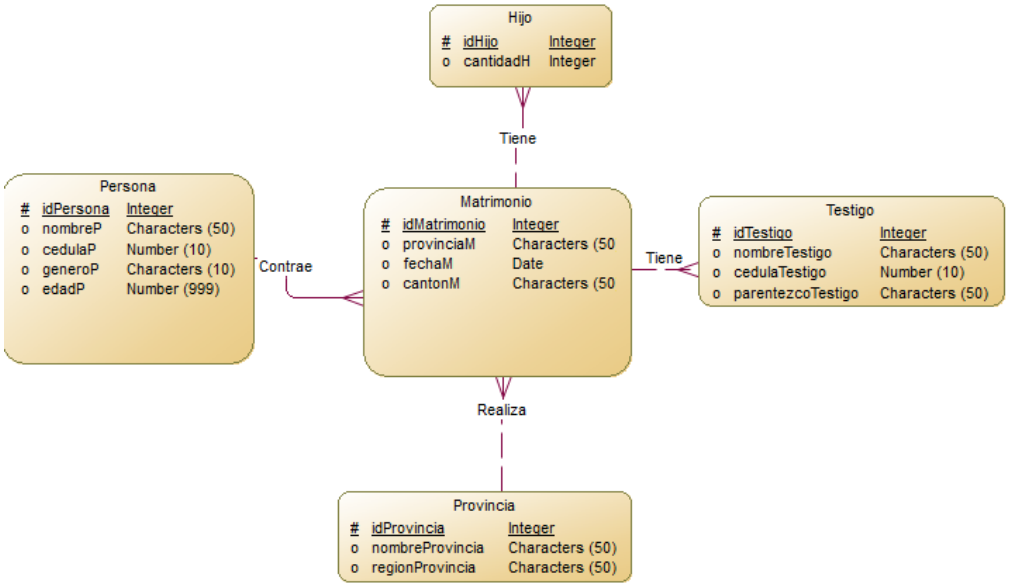


2. Planteamiento del esquema del ejercicio en Power Designer

Una vez obtenidas las tablas se las plantean en la herramienta Power Designer para tener una imagen gráfica de sus representaciones.

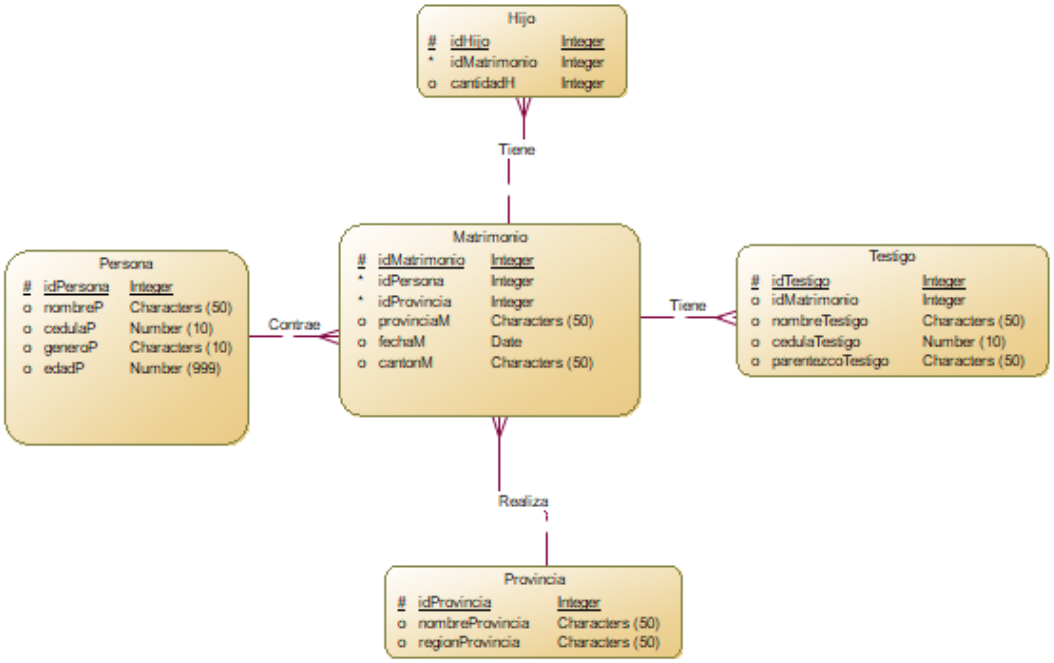
Modelo conceptual matrimonio.

Figura 28. Modelo conceptual en la herramienta Power Designer



Modelo lógico matrimonio.

Figura 29. Modelo lógico en la herramienta Power Designer



Modelo físico matrimonio.

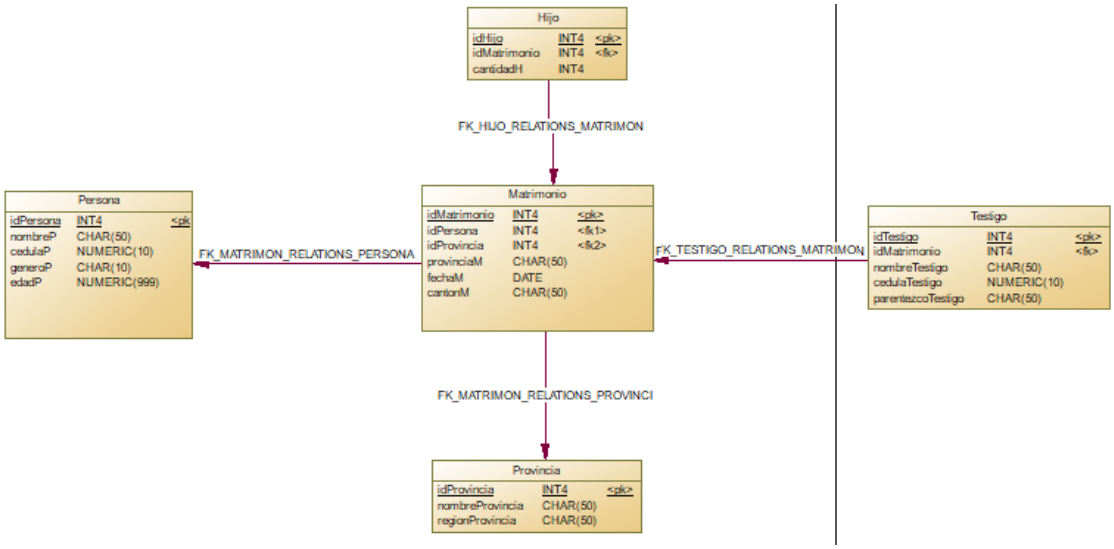


Figura 30. Modelo físico en la herramienta Power Designer

3.1.2 Orígenes SQL

SQL tiene sus orígenes en un trabajo realizado en el laboratorio de investigación de IBM en San José, California, a principios de los años setenta (Melton y Simon, 1993). Fue allí donde se creó un prototipo de implementación de los conceptos relacionales al que llamaron “System/R”. Este primer SGBD relacional incorporaba un lenguaje al que luego se conoció como SEQUEL; posteriormente, y por motivos legales, se le cambió el nombre por SQL. Esta es la razón por la que mucha gente todavía se refiere al lenguaje SQL como “sequel”. Desde 1973 hasta 1979, los investigadores de IBM publicaron en revistas académicas una gran cantidad de trabajos sobre el desarrollo del System/R. Ese periodo se caracterizó por una intensa discusión sobre la validez del SGBD relacional, en las conferencias y seminarios que tuvieron lugar tanto en los Estados Unidos como en Europa. Sin embargo, IBM tardó bastante en darse cuenta de la importancia comercial de los sistemas relacionales. Tuvo que ser ORACLE Corporation, fundada en 1977, quien explotara en el mundo comercial, por primera vez y con mucho éxito, las ideas subyacentes en el modelo de datos relacional.

3.1.3 ORACLE

ORACLE fue, y en la actualidad es, un SGBD Relacional basado en SQL. Por esta razón, en 1982 la Comisión Nacional de Estándares Americana encargó a su comisión de bases de datos (X3H2) que desarrollara un lenguaje de base de datos relacional estándar (RDL, Relational Database Language). En 1986 esta comisión proporcionó una definición de la sintaxis del SQL estándar, basándose principalmente en los dialectos de SQL de IBM y ORACLE. La Organización Internacional de Estándares (ISO) seguidamente lo adoptó en 1987 (ISO, 1987) con una publicación muy parecida del estándar. A este estándar se le conoce como SQL1. El documento original de ANSI especifica dos niveles para el SQL1: nivel uno y nivel dos. El nivel dos es el lenguaje SQL completo. El nivel uno es un subconjunto del nivel dos y se pretendía que originalmente actuara como una intersección de las implementaciones existentes.

3.1.4 Tipo de datos en SQL

Los tipos de datos SQL se clasifican en 13 tipos de datos primarios y de varios sinónimos válidos reconocidos por dichos tipos de datos. Los tipos de datos primarios son:

Tabla 14. Tipo de datos SQL

Tipo de Datos	Longitud	Descripción
BINARY	1 byte	Para consultas sobre tabla adjunta de productos de bases de datos que definen un tipo de datos Binario.
BIT	1 byte	Valores Si/No ó True/False
BYTE	1 byte	Un valor entero entre 0 y 255.
COUNTER	4 bytes	Un número incrementado automáticamente (de tipo Long)
CURRENCY	8 bytes	Un entero escalable entre 922.337.203.685.477,5808 y 922.337.203.685.477,5807.
DATETIME	8 bytes	Un valor de fecha u hora entre los años 100 y 9999.

SINGLE	4 bytes	Un valor en punto flotante de precisión simple con un rango de - 3.402823*1038 a -1.401298*10-45 para valores negativos, 1.401298*10- 45 a 3.402823*1038 para valores positivos, y 0.
DOUBLE	8 bytes	Un valor en punto flotante de doble precisión con un rango de - 1.79769313486232*10308 a -4.94065645841247*10-324 para valores negativos, 4.94065645841247*10-324 a 1.79769313486232*10308 para valores positivos, y 0.
SHORT	2 bytes	Un entero corto entre - 32,768 y 32,767.
LONG	4 bytes	Un entero largo entre - 2,147,483,648 y 2,147,483,647.
LONGTEXT	1 byte por carácter	De cero a un máximo de 1.2 gigabytes.
LONGBINARY	Según se necesite	De cero 1 gigabyte. Utilizado para objetos OLE.
TEXT	1 byte por carácter	De cero a 255 caracteres.

En estos tipos de datos se los puede clasificar por números exactos, números aproximados, hora y fecha, cadena de caracteres, cadena de caracteres *unicode*, cadenas binarias y otros tipos de datos. A continuación, se describe el tipo de datos que pertenecen a estos grupos, los intervalos, y su almacenamiento.

Tabla 15. Tipos de datos SQL 2

Grupo	Tipo de dato	Intervalo	Almacenamiento
Numéricos exactos	bigint	De -263 (-9.223.372.036.854.775.808) a 263 - 1 (9.223.372.036.854.775.807)	8 bytes
	int	De -231 (-2.147.483.648) a 231 - 1 (2.147.483.647)	4 bytes
	smallint	De -215 (-32.768) a 215 - 1 (32.767)	2 bytes
	tinyint	De 0 a 255	1 byte
	bit	Tipo de datos entero que puede aceptar los valores 1, 0 ó NULL	2 bytes
	decimal, numeric, decimal (p, s)	p (precisión): el número total máximo de dígitos decimales que se puede almacenar, tanto a la izquierda como a la derecha del separador decimal. La precisión	Precisión 1 - 9: 5 bytes

		debe ser un valor comprendido entre 1 y la precisión máxima de 38. La precisión predeterminada es 18. s (escala): el número máximo de dígitos decimales que se puede almacenar a la derecha del separador decimal. La escala debe ser un valor comprendido entre 0 y p. Solo es posible especificar la escala si se ha especificado la precisión. La escala predeterminada es 0. Con precisión máxima 1038 +1 y 1038 - 1	
	money	Tipos de datos que representan valores monetarios o de moneda: de -922.337.203.685,4775808 a 922.337.203.685,4775807	8 bytes
	smallmoney	De - 214,7483648 a 214,7483647	4 bytes
Numéricos aproximados	float	De - 1,79E+308 a -2,23E-308, 0 y de 2,23E-308 a 1,79E+308	Depende del valor de n
	real	De - 3,40E + 38 a -1,18E - 38, 0 y de 1,18E - 38 a 3,40E + 38	4 Bytes
Fecha y hora	datetime	Del 1 de enero de 1753 hasta el 31 de diciembre de 9999	
	smalldatetime	Del 1 de enero de 1900 hasta el 6 de junio de 2079	
Cadenas de caracteres	char (n)	Caracteres no Unicode de longitud fija, con una longitud de n bytes. n debe ser un valor entre 1 y 8.000	n bytes
	varchar (n)	Caracteres no Unicode de longitud variable. n indica que el tamaño de almacenamiento máximo es de 231 - 1 bytes	n bytes (aprox.)
	text	En desuso, sustituido por varchar. Datos no Unicode de longitud variable con una longitud máxima de 231 - 1 (2.147.483.647) caracteres	max bytes (aprox.)
Cadenas de caracteres unicode	nchar (n)	Datos de carácter Unicode de longitud fija, con n caracteres. n debe estar comprendido entre 1 y 4.000	2 * n bytes
	nvarchar (n)	Datos de carácter Unicode de longitud variable. n indica que el tamaño máximo de almacenamiento es 231 - 1 bytes	2 * n bytes + 2 bytes
	ntext (n)	En desuso, sustituido por nvarchar. Datos Unicode de longitud variable con una longitud máxima de 230 - 1 (1.073.741.823) caracteres	2 * n bytes
Cadenas binarias	binary (n)	Datos binarios de longitud fija con una longitud de n bytes, donde n es un valor que oscila entre 1 y 8.000	n bytes
	varbinary (n)	Datos binarios de longitud variable. n indica que el tamaño de almacenamiento máximo es de 231 - 1 bytes	n bytes
	image	En desuso, sustituido por varbinary. Datos binarios de longitud variable desde 0 hasta 231 - 1 (2.147.483.647) bytes	
Otros tipos de datos	cursor	Tipo de datos para las variables o para los parámetros de resultado de los procedimientos almacenados que contiene una referencia a un cursor. Las variables creadas con el tipo de datos cursor aceptan NULL	
	timestamp	Tipo de datos que expone números binarios únicos generados automáticamente en una base de datos. El tipo de datos timestamp es simplemente un número que se incrementa y no conserva una fecha o una hora	8 bytes
	sql_variant	Tipo de datos que almacena valores de varios tipos de datos aceptados en SQL Server, excepto text, ntext, image, timestamp y sql_variant	
	uniqueidentifier	Es un GUID (Globally Unique Identifier, Identificador Único Global)	16 bytes
	table	Es un tipo de datos especial que se puede utilizar para almacenar un conjunto de resultados para su procesamiento posterior. table se utiliza principalmente	

		para el almacenamiento temporal de un conjunto de filas devuelto como el conjunto de resultados de una función con valores de tabla
	xml	Almacena datos de XML. Puede almacenar instancias de xml en una columna o una variable de tipo xml

A continuación, se muestra el tamaño y formato de los tipos de datos DATE, TIME, DATETIME y TIMESTAMP

Tabla 16. Tipos de datos SQL 3

Tipo de datos	Tamaño	Formato
DATE	3 bytes	**YYYY-MM-DD**
TIME	5 bytes	**hh:mm:ss:nnnnnnn**
DATETIME	8 bytes	**YYYY-MM-DD hh:mm:ss:nnn**
TIMESTAMP	4 bytes	**YYYY-MM-DD hh:mm:ss**

En resumen, de las tablas anteriores se expresa una tabla donde muestra cuales son las principales diferencias entre algunos tipos de datos SQL.

Tabla 17. Tipos de datos SQL 3

Tipo de datos	Carácter	Almacenamiento
CHAR	UTF-8 variable	1 byte
VARCHAR	UTF-8 variable	1 byte
NCHAR	UTF-16	2 byte
NVARCHAR	UTF-16	2byte

3.2 Manipulación de la base de datos

Un lenguaje de manipulación de datos o (DML, Data Manipulation Languaje), son una serie de instrucciones que modifican los datos guardados en los objetos de las bases de datos (las tablas). Esta categoría contiene las instrucciones de INSERT, UPDATE y DELETE.

3.2.1 Definición de CRUD

CRUD cuya son las siglas en inglés de Create, Read, Update y Delete (crear, leer, actualizar y eliminar), que son las letras utilizadas en el cuerpo del diagrama. El concepto de la matriz CRUD es muy sencillo: un eje de la matriz representa los procesos principales del sistema de aplicaciones. El otro eje designa a las entidades principales utilizadas por el sistema de aplicaciones. En cada celda de la matriz, se

escribe la combinación adecuada de letras las cuales determinan ciertas actividades tales como:

C, si el proceso crea ocurrencias nuevas de la entidad.
R, si el proceso lee información sobre una entidad desde un origen de datos.
U, si el proceso actualiza uno o más atributos para la entidad.
D, si el proceso elimina ocurrencias de la entidad.

3.2.2 Realización de tablas aplicando instrucciones Insert, Update y Delete.

3.2.2.1 Creación de tablas

Tabla partido

Tabla 18. Tabla partido para utilizar sentencias de inserción

```
1   create table partido
2   (
3   id serial,
4   idequipo1 integer,
5   idequipo2 integer,
6   golesEquipo1 integer,
7   golesEquipo2 integer,
8   fechaPartido date,
9   constraint pk_partido primary key (id)
10  );
```

Tabla equipo

Tabla 19. Tabla equipo para utilizar sentencias de inserción

```
1   create table equipo
2   (
3   id integer,
4   descripcion character varying ,
5   partidosJugados integer,
6   puntos integer,
7   golDiferencia integer,
8   constraint pk_equipo primary key (id)
9   );
```

3.2.2.2 Ejecución de sentencias

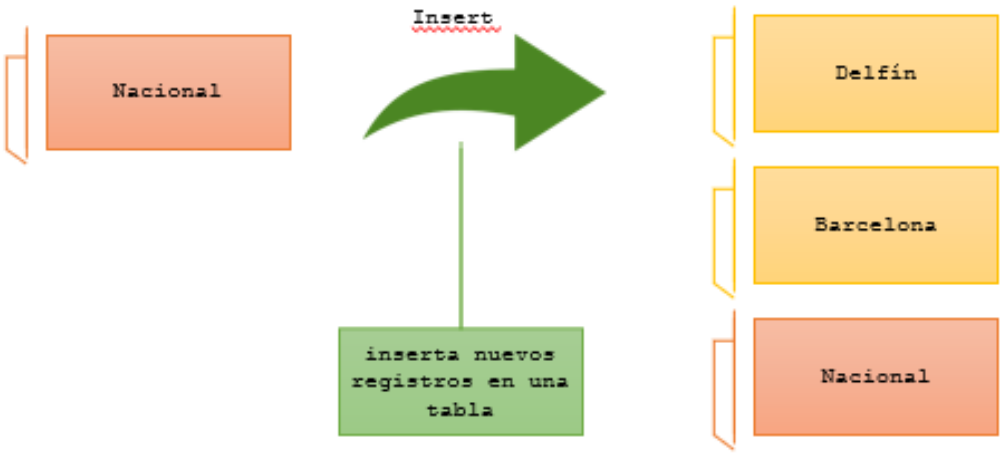
Existen tipos de órdenes de manipulación están diseñados para producir cambios en el propio estado de la base de datos como son los siguientes:

***Insert*:** Permite crear o insertar nuevos registros en una tabla.

Tabla 20. Uso de sentencias de inserción en la tabla equipo

```
1  insert into equipo
2  (id, descripcion, partidosJugados,
3   puntos, golDiferencia)
4  values (1, 'delfin',5,9,6),
5  (2, 'barcelona',5,9,6),
6  (3, 'nacional',6,15,2);
```

Figura 31. Naturaleza set-at-a-time de consulta SQL Insert



Resultado obtenido:

Tabla 21. Resultado de operación de inserción en la tabla equipo

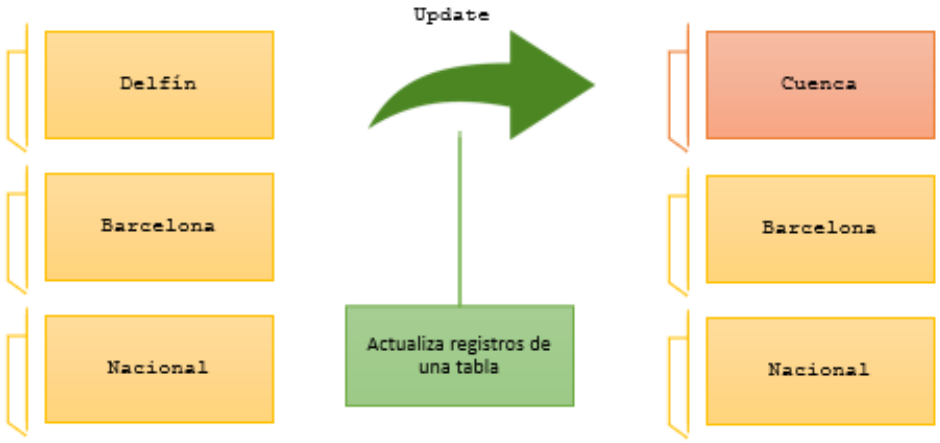
Id (integer)	Descripción (carácter varying)	partidosJugados (integer)	Puntos (integer)	golDiferencia (integer)
1	Delfin	5	9	6
2	Barcelona	5	9	6
3	Nacional	6	15	2

Update: Permite actualizar registros de una tabla.

Tabla 22. Sentencia SQL de actualización en la tabla equipo

```
1  update equipo set
2  descripcion='cuenca'
3   where id=1;
```

Figura 32. Naturaleza set-at-a-time de consulta SQL Update



Resultado obtenido:

Tabla 23. Resultado de la operación de actualización en la tabla equipo

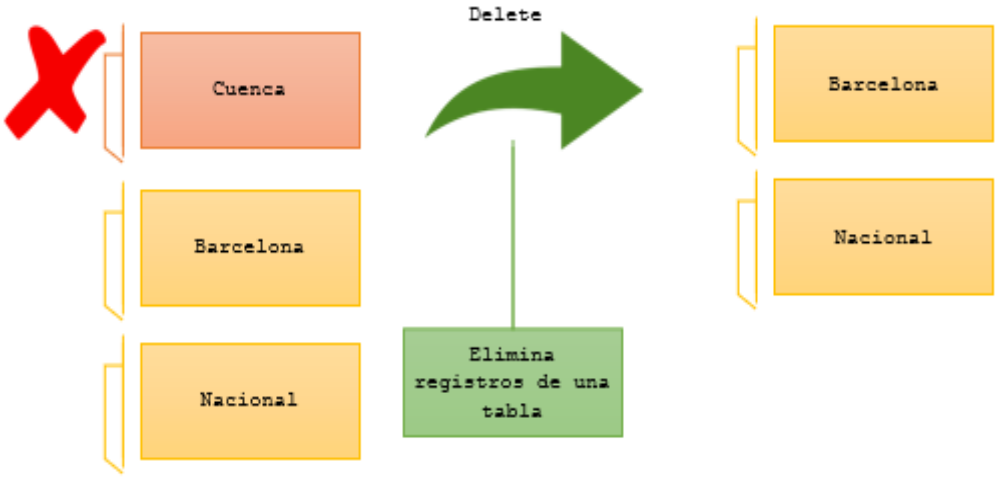
Id (integer)	Descripción (carácter varying)	partidosJugados (integer)	Puntos (integer)	golDiferencia (integer)
1	Cuenca	5	9	6
2	Barcelona	5	9	6
3	Nacional	6	15	2

Delete: Permite eliminar registros de una tabla.

Tabla 24. Sentencia SQL de eliminación en la tabla equipo

```
1  delete from
2  equipo where
3  id = 1
```

Figura 33. Naturaleza set-at-a-time de consulta SQL Eliminación



Resultado obtenido:

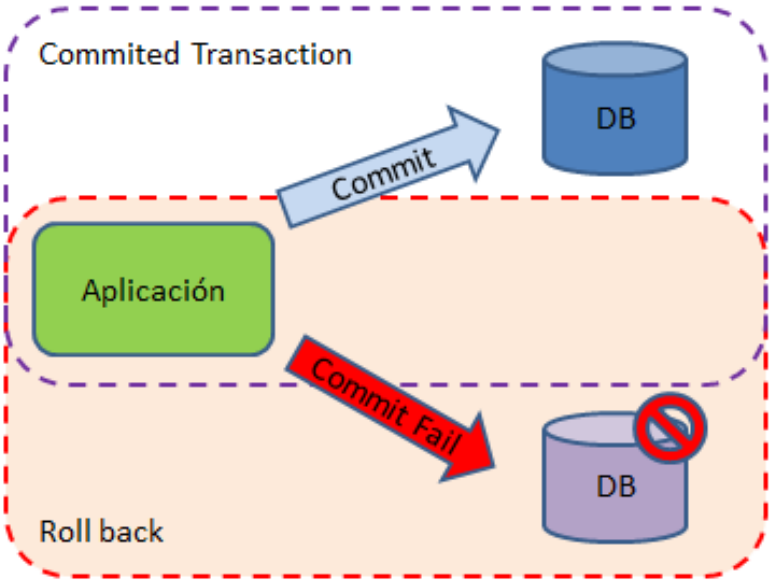
Tabla 25. Resultado de la operación de eliminación en la tabla equipo

Id (integer)	Descripción (carácter varying)	partidosJugados (integer)	Puntos (integer)	golDiferencia (integer)
2	Barcelona	5	9	6
3	Nacional	6	15	2

3.3 Gestión de transacciones.

Un sistema de información de Base de Datos relacionales (RDBMS), es una transacción es una serie de una o más instrucciones SQL tratadas como una sola unidad. Una transacción debe funcionar o fracasar por completo, lo que significa que los cambios que hace una transacción en cualquier base de datos deben volverse permanentes cuando la transacción concluye correctamente. Por otra parte, estos cambios deben eliminarse por completo de la base de datos si la transacción falla antes de su conclusión.

Figura 34. Transacciones (Oscar Blancarte,2014)



3.3.1 Transacciones

Son los eventos que causan un cambio de estado. Una transacción cambia una base de datos de un estado a otro. Se alcanza un nuevo estado validando los hechos que se han convertido en ciertos y/o negando los hechos que han dejado de ser verdad.

Una transacción es una unidad lógica de trabajo. O informalmente, y trabajando con SQL, un conjunto de sentencias que se ejecutan como si fuesen una sola. En general, las sentencias que forman parte de una transacción se interrelacionan entre sí, y no tiene sentido que se ejecute una sin que se ejecuten las demás.

La mayoría de las transacciones se inician de forma implícita al utilizar alguna sentencia que empieza con CREATE, ALTER, DROP, SET, DECLARE, GRANT o REVOKE, aunque existe la sentencia SQL para iniciar transacciones, que es la siguiente:

Figura 35. Acceso al modo transaccional en un gestor de bases de datos

```
Set transaction read write;
```

Una transacción siempre debe acabar explícitamente con alguna de las sentencias *COMMIT O ROLLBAC*K, su principal diferencia es que *COMMIT* confirma todos los

cambios realizados en la base de datos, la sentencia *ROLLBACK* deshace todos los cambios que se hayan producido en la base de datos y lo deja todo como estaba antes que se produjera alguna transacción.

Figura 36. Ejemplo de una transacción

```
SET TRANSACTION READ WRITE;
UPDATE empleados SET sueldo = sueldo - 1000 WHERE num_proyec = 3;
UPDATE empleados SET sueldo = sueldo + 1000 WHERE num_proyec = 1;
COMMIT;
```

3.3.2 Propiedades de los sistemas gestores de base de datos.

Atomicidad: es la propiedad que se asegura que la operación se ha realizado o no, y ante un fallo del sistema no puede quedar a medias.

Consistencia: es la propiedad que se asegura que solo se empieza aquello que se puede acabar, se ejecutan operaciones que no va a romper las reglas o directrices.

Aislamiento: es la propiedad que se asegura que una operación no puede afectar a otras. Por ende, asegura que la realización de dos transacciones sobre la misma información sea independiente.

Durabilidad. Es la propiedad que se asegura que, una vez realizada la operación, esta persistirá, aunque falle el sistema.

3.3.3 Ejemplo de ejecución de transacciones con las propiedades de los sistemas gestores de base de datos.

3.3.3.1 Creación de tablas

Tabla Crédito

Tabla 26. Sentencia SQL de tabla cuenta para prueba de transacciones

```
1  create table CUENTAS (N_CUENTA numeric(8,0) ,
2                        NOMBRE date,
3                        BALANCE numeric(8,0))
```

Comando COMMIT: Confirma que todas las sentencias son correctas.

Tabla 27. Conjunto de sentencias individuales evaluadas en un bloque COMMIT para su consideración como transacción

```
1  INSERT INTO CUENTAS (N_CUENTA, NOMBRE, BALANCE) VALUES
2  (0679259, 'PEPE', 200);
3
4  UPDATE CUENTAS SET BALANCE = BALANCE - 137.00 WHERE NOMBRE
5  = 'PEPE';
6
7  UPDATE CUENTAS SET BALANCE = BALANCE + 137.00 WHERE NOMBRE
8  = 'JUAN';
9
10 SELECT NOMBRE, BALANCE FROM CUENTAS WHERE NOMBRE = 'PEPE'
11 AND NOMBRE = 'JUAN';
12
13 COMMIT;
```

Comando BEGIN: Permite que se ejecuten todas las secuencias SQL que se necesite.

Tabla 28. Uso de la cláusula BEGIN para iniciar una transacción

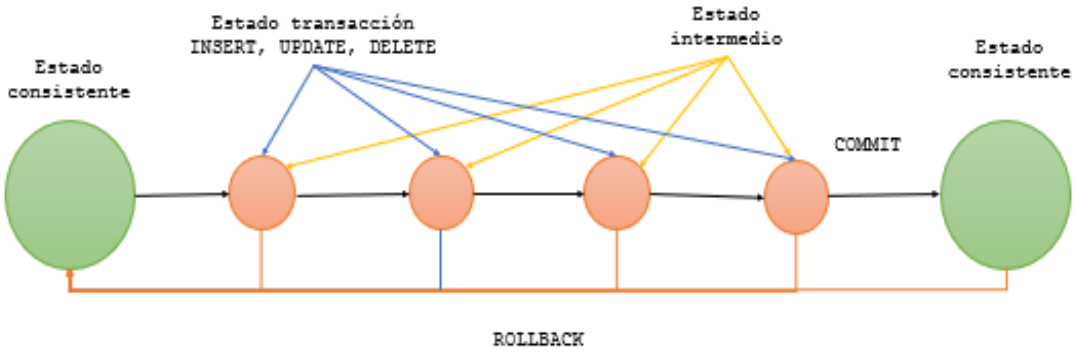
```
1  BEGIN;
2
3  UPDATE CUENTAS SET BALANCE = BALANCE - 100.00 WHERE N_CUENTA
4  = 0127365;
5
6  UPDATE CUENTAS SET BALANCE = BALANCE + 100.00 WHERE N_CUENTA
7  = 0795417;
```

Comando ROLLBACK: Deshace los cambios o las transacciones realizadas.

Tabla 29. Uso de la cláusula ROLLBACK para deshacer cambios de un conjunto de sentencias de una transacción

```
1  BEGIN;
2  "SENTENCIAS SQL"
3  COMMIT;
4   ROLLBACK;
```

Figura 37. Esquema del funcionamiento de una transacción



3.4 Estructura básica de las consultas SQL.

SQL (Lenguaje de consulta estructurado) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas. Una de sus características es el manejo del álgebra y el cálculo relacional permitiendo lanzar consultas con el fin de recuperar -de una forma sencilla- información de interés de una base de datos, así como también hacer cambios sobre la misma.

En los siguientes párrafos se efectuarán consultas con sus determinadas sentencias con la base de datos del sistema NORTHWIND.

3.4.1 Sentencias

SELECT: Palabra clave que indica que la sentencia de SQL que queremos ejecutar es de selección.

Selecciona el número de empleados por título.

Tabla 30. Forma general de la sentencia SELECT

```
1  select title, max(Employeeid)
2  from Employees group by title
3  order by title asc;
```

Resultado

Tabla 31. Resultado de una operación SELECT en la tabla Employees

	Title Character varying (30)	Max smallint
1	Inside Sales Coordinator	8
2	Sales Manager	5
3	Sales Representative	9
4	Vice President. Sales	2

ALL: Indica que queremos seleccionar todos los valores. Es el valor por defecto y no suele especificarse casi nunca.

DISTINCT: Indica que queremos seleccionar sólo los valores distintos.

Tabla 32. Forma general del uso de la cláusula Distinct

```
1  select distinct
2  region from employees;
```

Resultado

Tabla 33. Resultado del uso de la cláusula Distinct en la tabla employees

	region Character varying (25)
1	Null
2	WA

FROM: Indica la tabla (o tablas) desde la que queremos recuperar los datos. En el caso de que exista más de una tabla se denomina a la consulta "consulta combinada" o "join". En las consultas combinadas es necesario aplicar una condición de combinación a través de una cláusula WHERE.

Tabla 34. Forma general del uso de la cláusula From en una operación SELECT

```
1  select Regionid,
2  count(*) from Region
3  group by Regionid
```

Resultado

Tabla 35. Resultado de un select-from en la tabla Región

	Title Character varying (30)	Max Smallint
1	Inside Sales Coordinator	8
2	Sales Manager	5
3	Sales Representative	9
4	Vice President. Sales	2

WHERE: Especifica una condición que debe cumplirse para que los datos sean devueltos por la consulta. Admite los operadores lógicos AND y OR.

Tabla 36. Forma general de la cláusula Where en una consulta SQL

```
1  select EmployeeID,
2  count(EmployeeID)
3  from employeeTerritories
4  where EmployeeID='2'
5  group by EmployeeID
```

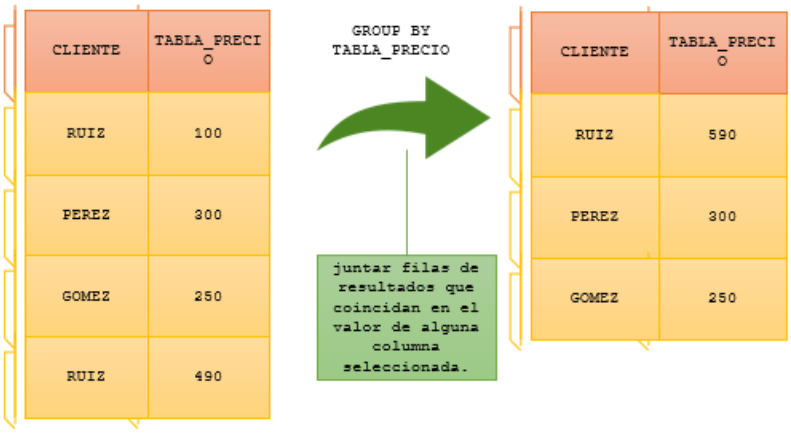
Resultado

Tabla 37. Resultado de las cláusulas select-from-where en la tabla employeeTerritories

	employeeid smallint	Count Bigint
1	2	7

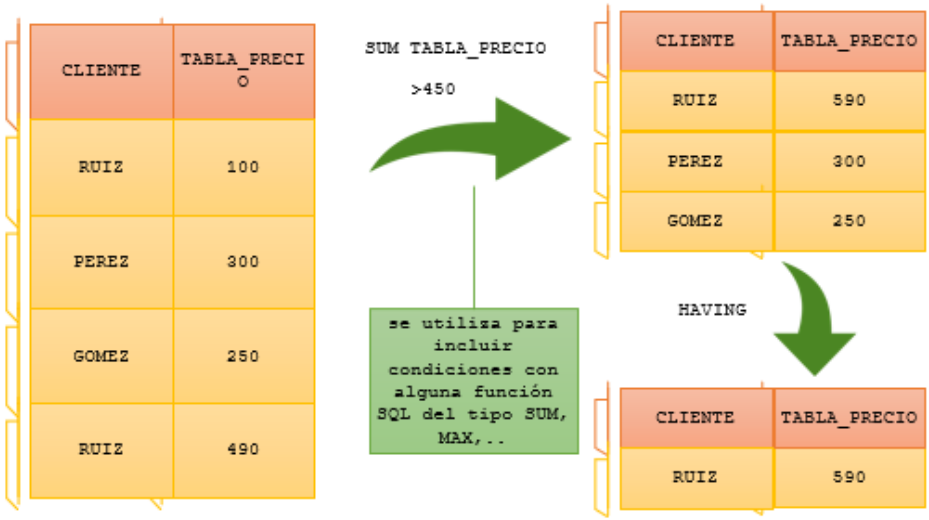
GROUP BY: Especifica la agrupación que se da a los datos. Se usa siempre en combinación con funciones agregadas.

Figura 38. Funcionamiento de la cláusula Group By de SQL



HAVING: Especifica una condición que debe cumplirse para los datos. Especifica una condición que debe cumplirse para que los datos sean devueltos por la consulta. Su funcionamiento es similar al de WHERE pero aplicado al conjunto de resultados devueltos por la consulta. Debe aplicarse siempre junto a GROUP BY y la condición debe estar referida a los campos contenidos en ella.

Figura 39. Funcionamiento de la operación de agregación Having de SQL



ORDER BY: Presenta el resultado ordenado por las columnas indicadas. El orden puede expresarse con ASC (orden ascendente) y DESC (orden descendente). El valor predeterminado es ASC.

Tabla 38. Operación select con cláusula group by

```
1  select title,max(Employeeid)
2  from Employees group by title
3  order by title asc;
```

Resultado

Tabla 39. Resultado de operación select group by de sql

	Title Character varying (30)	max smallint
1	Inside Sales Coordinator	8
2	Sales Manager	5
3	Sales Representative	9
4	Vice President, Sales	2

3.5 Funciones de agregación.

El estándar define cinco funciones agregadas:

- • ***COUNT.*** Devuelve el número de valores que hay en una columna.

Figura 40. Funcionamiento de la operación de agregación count

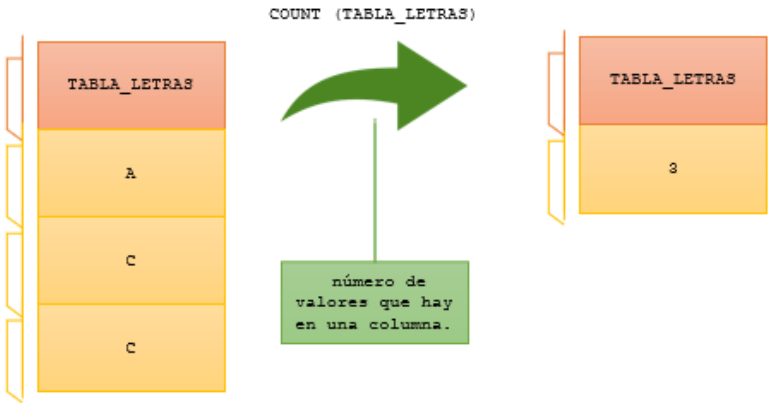


Tabla 40. Sentencia SQL con el uso de la operación de agregación count

```
1  select count(*) as FilasTotales,
2   count(PostalCode) as CodigoPostalTotal
3   from Customers
```

Resultado

Tabla 41. Resultado de la sentencia sql con la función de agregación count

	Filatotales bignit	codigopostaltotal
1	91	80

• ***SUM***. Devuelve la suma de los valores de una columna.

Figura 41. Funcionamiento de la operación de agregación sum

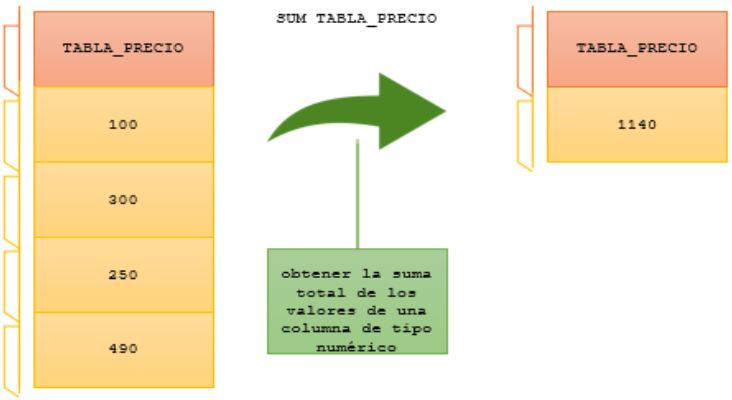


Tabla 42. Sentencia sql con la función de agregación count

```
1  select count(*) as
2  FilasTotales, count(PostalCode)
3  as CodigoPostalTotal from Customers
```

Resultado

Tabla 43. Resultado de sentencia sql con la función de agregación count

	Country character varying (15)	employeeid	Totalsumatoria bigint
1	USA	1	1
2	USA	2	2
3	USA	3	3
4	USA	4	4
5	USA	8	8

• **AVG.** Devuelve la media de los valores de una columna.

Figura 42. Funcionamiento de la función de agregación avg

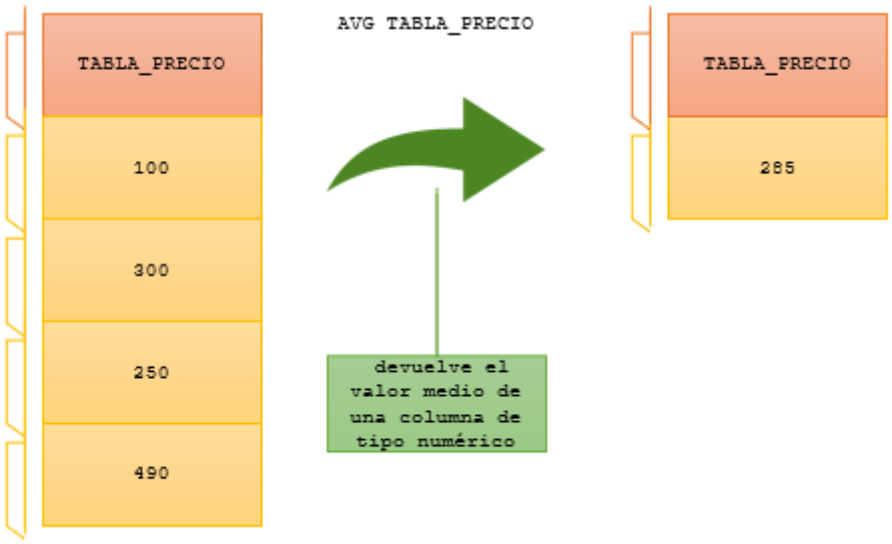


Tabla 44. Sentencia sql con la función de agregación avg

```
1  select avg(UnitPrice)as
2   Total from Order_Details
```

Resultado

Tabla 45. Resultado de sentencia sql con la función de agregación avg

	Total double precision
1	26.21851971102221

MIN. Devuelve el valor mínimo de un conjunto de valores en una columna.

Figura 43. Funcionamiento de la función min

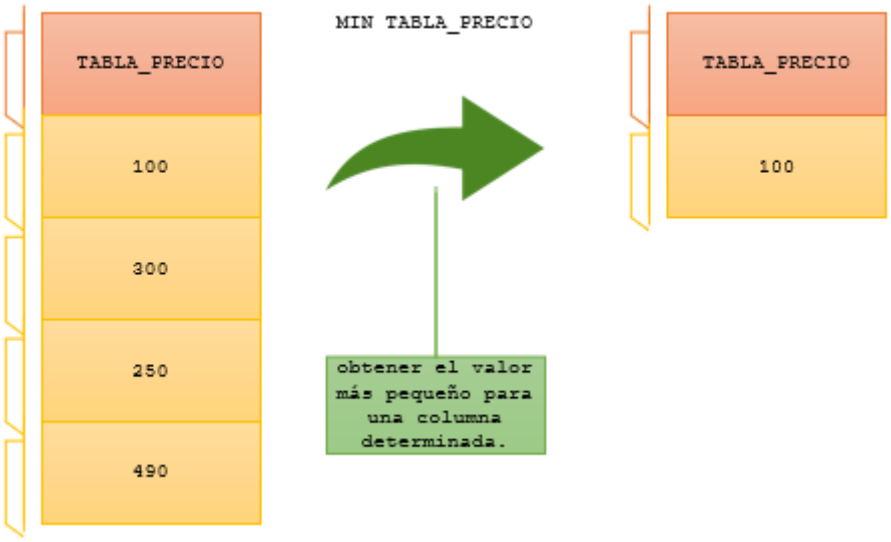


Tabla 46. Sentencia sql con el uso de la función min

```
1  select title,
2  min(Employeeid) from Employees
3  group by title order by title asc;
```

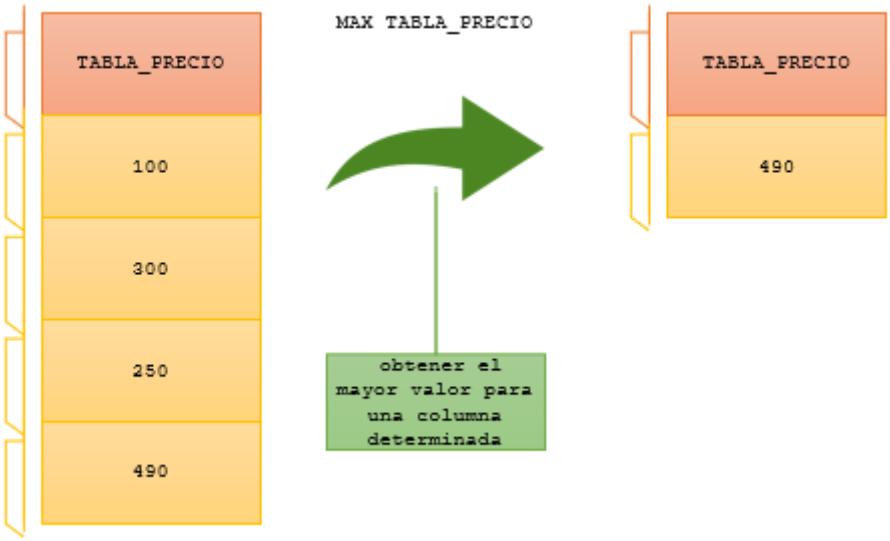
Resultado

Tabla 47. Resultado de la sentencia sql con el uso de la función min

	Title Character varying (30)	max smallint
1	Inside Sales Coordinator	8
2	Sales Manager	5
3	Sales Representative	1
4	Vice President, Sales	2

• ***MAX.*** Devuelve el valor máximo del conjunto de valores en una columna.

Figura 44. Funcionamiento de la función max



3.6 Subconsultas y consultas correlacionadas.

La palabra estructurado, en el Lenguaje de Consulta Estructurado, originalmente se refería a la capacidad para anidar consultas en las sentencias select. Una consulta anidada se denomina "subconsulta". Las subconsultas pueden anidarse en un número de niveles arbitrario.

SQL evalúa las subconsultas en orden desde la consulta más interna a la más externa.

El resultado desde la consulta más interna se pasa por valor a la más externa.

Podemos distinguir tres tipos de subconsultas según el tipo de resultado que devuelve la subconsulta:

- • Subconsulta escalar. Devuelve un único valor (intersección de fila/columna) como resultado.

Por lo tanto, las subconsultas escalares tienden a usarse con operadores como >, <, <>, =.

- • Subconsulta de fila. Devuelve múltiples columnas procedentes de una única fila.

- • Subconsulta de tabla. Devuelve como resultado una tabla (múltiples filas y columnas).

Una tabla subconsulta puede usarse con un operador IN. IN espera recibir un conjunto de valores.

Subconsultas correlacionadas

En los ejemplos anteriores, la subconsulta se ejecutó una vez. El valor resultante se utilizó después en la cláusula *where* de la consulta más externa. Sin embargo, una subconsulta puede ejecutarse repetidamente. En este caso produce una serie de valores que van a ser comparados con los resultados obtenidos por la consulta más externa.

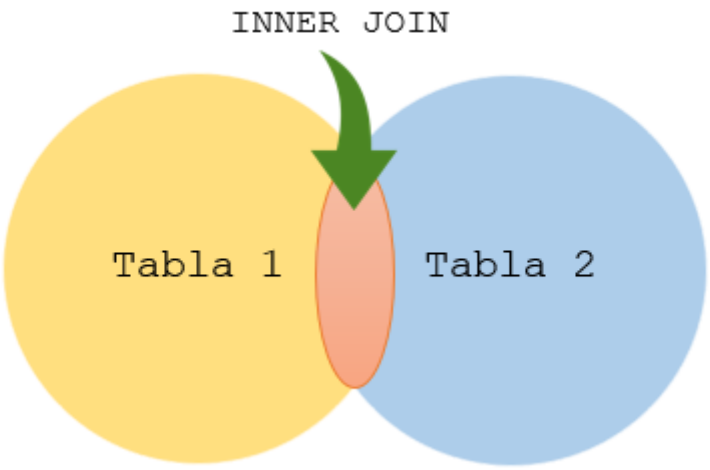
3.7 Operadores de reunión en SQL.

La razón principal para elaborar SQL2 (ISO, 1992) como un sustituto del SQL1 ha sido eliminar las restricciones innecesarias y generalizar al máximo las operaciones dentro del sublenguaje de la base de datos. Esto se hace evidente en la forma en la que en el nuevo estándar se usan los operadores del álgebra relacional *join, union, intersect y except* (diferencia).

Ejemplo

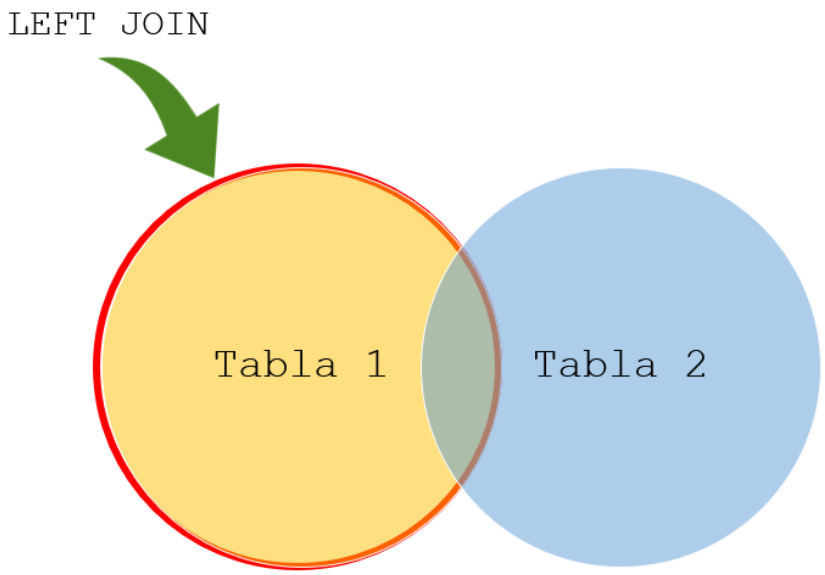
INNER JOIN: Combina los registros de dos tablas si hay valores coincidentes en un campo común.

Figura 45. Descripción de las cláusulas JOIN de SQL



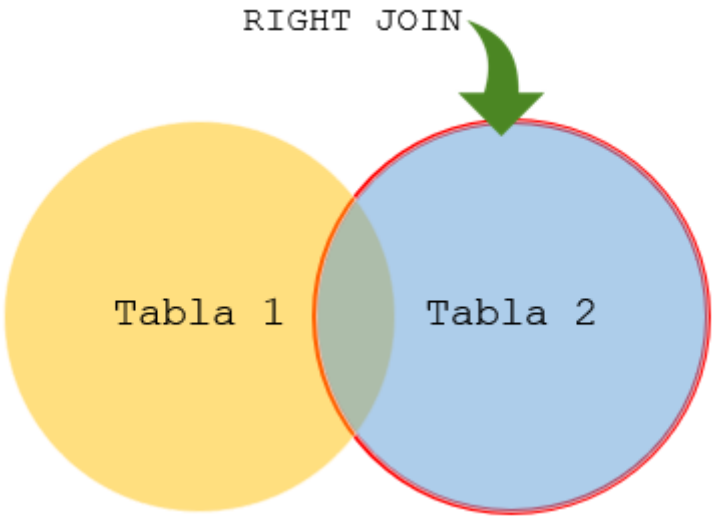
LEFT JOIN: devuelve todos los registros de la tabla izquierda y los registros coincidentes de la tabla derecha.

Figura 46. Descripción de la cláusula Left Join de SQL



RIGHT JOIN: devuelve todos los registros de la tabla derecha y los registros coincidentes de la tabla izquierda.

Figura 47. Descripción de la cláusula Left Join de SQL



FULL JOIN: devuelve todos los registros cuando hay una coincidencia en la tabla izquierda o derecha.

Figura 48. Descripción de la cláusula full join de SQL

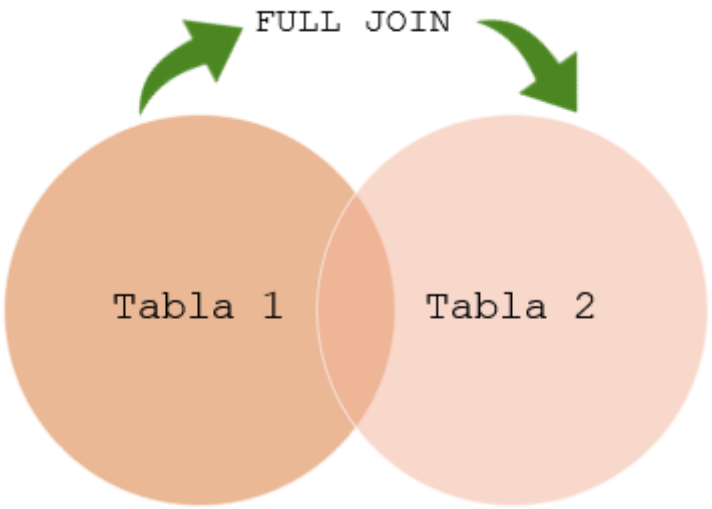


Tabla 48. Sentencia SQL con el uso de natural join

```
1  SELECT *FROM TERRITORIES
2  NATURAL JOIN
3   REGION
```

Resultado

Tabla 49. Resultado de una consulta sql utilizando natural join

	Regionid smallint	Territoryid Character varying (20)	Territorydescription Character	regiondescription Character
1	1	01581	Westboro	Eastern
2	1	01730	Bedford	Eastern
3	1	01833	Georgetow	Eastern
4	1	02116	Boston	Eastern
5	1	02139	Cambridge	Eastern
….53	2	98104	Seattle	Western

UNION: Permite combinar los resultados de varias instrucciones SELECT en un único conjunto de resultados.

Figura 49. Descripción de la operación union de sql

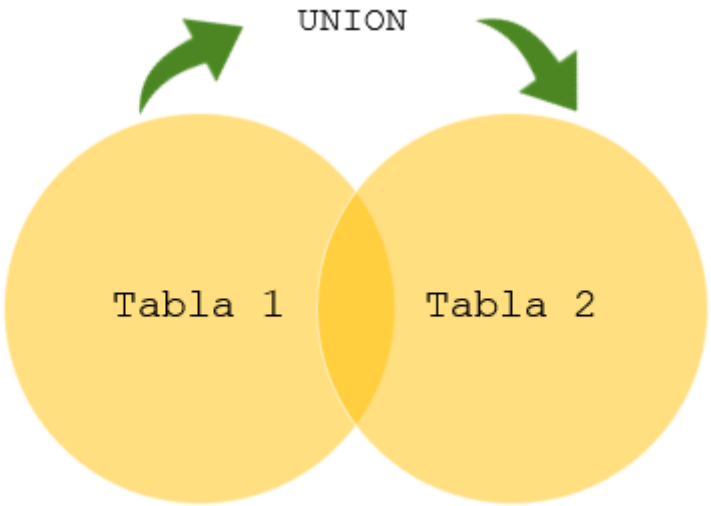


Tabla 50. Sentencia sql con el uso de la operación union

```
1  select lastname, city from employees
2   where lastname ='Fuller' UNION select customerid,
3  shipcountry from orders where customerid ='King'
```

Resultado

Tabla 51. Resultado de la operación union de sql

	Lastname Character varying +	city Character varying (15)
1	Fuller	Tacoma

INTERSECT: Devuelve los valores distintos devueltos por las consultas y comunes a ambas, con lo que obtenemos una intersección.

Figura 50. Descripción de la operación intersect de sql

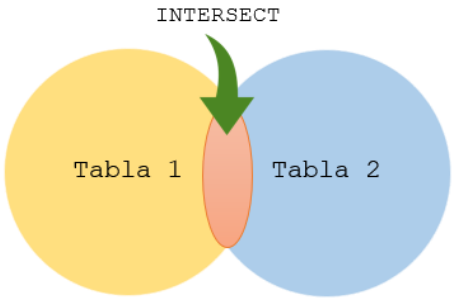


Tabla 52. Sentencia sql con el uso de la operación intersect de sql

```
1  select region from Suppliers
2  intersect select region
3  from Customers
```

Resultado

Tabla 53. Resultado de la operación intersect de sql

	region **Character varying (15)**
1	Null
2	Québec
3	OR

3.8 Cuestionario de la unidad

1. ¿Qué es una entidad?

Cualquier cosa (persona, lugar, cosa o hecho).

Una característica de una entidad.

Dependiendo del interés del sistema de información.

Características generales.

2. ¿Cuáles son los elementos básicos de todos los modelos de datos?

Entidades, atributos, relaciones y restricciones.

Correspondencia, generalización, especialización.

Herramienta de base de datos y niveles de abstracción.

Los establecidos por el departamento de TICs.

3. ¿Cuál es la diferencia entre un lenguaje de consulta de un DBMS y un lenguaje de procedimiento?

Un lenguaje de consulta permite especificar al usuario qué debe hacerse sin tener que especificar cómo debe hacerse.

Un lenguaje de procedimiento permite especificar al usuario qué debe hacerse sin tener que especificar cómo debe hacerse.

La diferencia es relativa a la tecnología que se usa.

Depende de los comandos que se apliquen.

4. ¿Qué tipo de lenguaje es considerado SQL?

Declarativo.

Procedimental.

Estructurado.

Orientado a objetos.

5. Es la propiedad que asegura que la operación se ha realizado o no, por lo tanto, ante un fallo del sistema no puede quedar a medias.

Atomicidad

Consistencia

Aislamiento

Durabilidad

6. Es la propiedad que asegura que solo se empieza aquello que se puede acabar. Por lo tanto, se ejecutan aquellas operaciones que no van a romper las reglas y directrices

Consistencia

Atomicidad

Aislamiento

Durabilidad

7. Es la propiedad que asegura que una operación no puede afectar a otras. Esto asegura que la realización de dos transacciones sobre la misma información, sean independientes y no generen ningún tipo de error.

Aislamiento

Atomicidad

Consistencia

Durabilidad

8. Es la propiedad que asegura que, una vez realizada la operación, está persistirá y no se podrá deshacer, aunque falle el sistema.

Durabilidad

Atomicidad

Consistencia

Aislamiento

9. ¿Qué funciones de agregación estable el estándar SQL?

COUNT, SUM, AVG, MIN, MAX

CREATE, ALTER, DROP

SELECT, INSERT, UPDATE, DELETE

GRANT, REVOKE

10. ¿A qué se refiere el término "subconsulta" en SQL?

A la capacidad para anidar consultas en las sentencias select.

A la capacidad de utilizar predicados en la cláusula where.

A la capacidad de realizar operaciones de algebra relacional entre relaciones.

A la capacidad de utilizar JOINS

BIBLIOGRAFÍA

Abelló, A., Rollón, E. y Rodríguez, M. E. (2006). *Diseño y administración de bases de datos.* U. P. de Catalunya. https://elibro.net/es/ereader/uleam/61394?page=1

Abelló, A., Rollón, E. y Rodríguez, M. E. (2006). *Diseño y administración de bases de datos.* U. P. de Catalunya. https://elibro.net/es/ereader/uleam/61394?page=1

Coronel, C., Morris, S. y Rob, P. (2011). *BASES DE DATOS: Diseño, implementación y administración.* (Novena). México.

Glenn Brookshear, J. (2012). *Introducción a la computación* (11a ed.). Person.

Jiménez Capel, M. Y. (2014). *Bases de datos relacionales y modelado de datos (UF1471)* (Primera). I. Editorial. https://elibro.net/es/ereader/uleam/106467?page=1

Marqués, M. (2011). *Bases de Datos* (Primera). https://elibro.net/es/ereader/uleam/51645?page=1

Medina Serrano, S. (2015). *SQL Server 2014, Soluciones prácticas de administración.* A-MA. https://elibro.net/es/ereader/uleam/106467?page=1

Oppel, A. (2010). *Fundamentos de Bases de Datos.* M. G. Hill. https://elibro.net/es/ereader/uleam/37322?page=1

Piñeiro Gómez, J. M. (2011). *Manual Gestión de bases de datos.* CEP. https://ebookcentral.proquest.com/lib/uleamecsp/reader.action?docID=3214275&ppg=3

Valderrey Sanz, P. (2014). *GESTIÓN DE BASES DE DATOS*. https://elibro.net/es/ereader/uleam/62469?page=1

SOBRE LOS AUTORES

ROBERT WILFRIDO MOREIRA CENTENO

Nació en Ecuador en 1983, recibió el título de Ingeniero en Sistemas Informáticos en la Universidad Técnica de Manabí en el 2009. A nivel de postgrado tiene el título de Magíster en Sistemas de Información Gerencial en la Escuela Superior Politécnica del Litoral en el año 2013. Se ha desempeñado como docente en la Universidad Técnica de Manabí, Universidad Estatal del Sur de Manabí y Universidad Laica Eloy Alfaro de Manabí en donde actualmente trabaja, impartiendo las cátedras de Programación Orientada a Objetos, Estructuras de Datos, Análisis, Diseño y Administración de Bases de Datos.

EDISON ERNESTO ALMEIDA ZAMBRANO

Profesor/Investigador de la Facultad Ciencias Informáticas de la Universidad Eloy Alfaro de Manabí, es Ingeniero en sistemas, Magíster en redes y telecomunicaciones, profesor en el uso de las tics en comunidades rurales, Coordinador de proyectos tecnológicos para la educación, entre los proyectos: Aulas tecnológicas comunitarias y Educar Ecuador (Plataforma Educativa e implementación de tecnología para los profesores de educación secundaria). Experto en la implementación de tecnología de bajo costo para zonas rurales, tiene experiencia en la creación, ejecución y monitoreo de proyectos tecnológicos para la educación e implementación de infraestructuras tecnológicas caso de estudio en la Pontificia Universidad Católica del Ecuador sede ciudad de Quito.

HOMERO RENÁN MENDOZA RODRÍGUEZ

Docente de la Facultad Ciencias Informáticas en la Universidad Laica Eloy Alfaro de Manabí (Uleam), con estudios de pre-grado y postgrado realizados en Hunter College, The City University of New York CUNY, con el título de Magíster en Matemáticas Puras. Recibió

una mención honoraria de National Honorary Mathematics Society y miembro de Pi Mu Epsilon Beta Chapter.

Tiene una amplia experiencia en la docencia de más de 10 años en instituciones como Maritime Academy for Science and Tecnology – Bronx, New York, Bushwick High School – Brooklyn, New York, CUNY Preparatory School – Bronx, New York y ULEAM – Manta, Ecuador. Producción científica con artículos publicados en Google Academic "Plan Informático para la Gestión Tecnológica", Proceedings of ICITS 2019 International Conference on Information Technology & Systems Springer: "Energy Consumption for Anti-virus Applications in Android OS." Ha participado en seminarios y congresos como ponente en la especialidad de Informática y la Comunicación.

ESTHELA MARÍA SAN ANDRÉS LAZ

Profesor Investigador de la Facultad de Ciencias Informáticas de la Universidad Técnica de Manabí, Doctor en Ciencias Pedagógicas, Máster en Informática de Gestión y Nuevas Tecnologías, Máster en Gerencia Educativa, docente de pregrado y posgrado, ha desempeñado varios cargos en instituciones de Educación Superior, Vicerrectora Académica en la Universidad Estatal Amazónica, Par Evaluador de Universidades y Escuelas Politécnicas del CACES, ha participado en procesos de autoevaluación en varias universidades, profesor revisor en varias revistas nacionales e internacionales, autora de varios libros y artículos en el ámbito de la innovación educativa e informática, ponente y conferencista en eventos nacionales e internacionales.

KATTY TATIANA MENDOZA MUÑOZ

Nacida en Ecuador, Manabí, Chone en el año de 1986, en el 2008 obtuvo el título de Licenciada en Ecoturismo y en el año 2017 se graduó de Licenciada en Ciencias de la Educación mención inglés, fue docente de la Pontificia Universidad Católica Sede Manabí, Instructor CEL Distrito 13D11 Zona 4, Profesor Centro Educativo CETECH, actualmente es Docente de la Unidad Educativa “Augusto Solórzano Hoyos”.

Uleam
UNIVERSIDAD LAICA
ELOY ALFARO DE MANABÍ

ISBN: 978-9942-827-76-0
9789942827760

2022