parte 1

Introducción a bases de datos

Página 2

Esta página se dejó en blanco intencionalmente

Página 3

capitulo 1

1

Bases de datos y Usuarios de la base de datos

Databases y sistemas de bases de datos son un elemento esencial componente de la vida en la sociedad moderna: la mayoría de nosotros encontrar varias actividades todos los días que implican alguna interacción con una base de datos. Por ejemplo, si vamos al banco a depositar o retirar fondos, si hacemos un hotel o reserva de aerolínea, si accedemos a un catálogo de biblioteca computarizado para buscar un artículo bibliográfico, o si compramos algo en línea, como un libro, un juguete o computadora: es probable que nuestras actividades involucren a alguien o alguna computadora programa que accede a una base de datos. Incluso comprar artículos en un supermercado a menudo actualiza automáticamente la base de datos que contiene el inventario de artículos comestibles.

Estas interacciones son ejemplos de lo que podemos llamar **base de datos tradicional. aplicaciones** , en las que la mayor parte de la información que se almacena y se accede a textual o numérico. En los últimos años, los avances en tecnología han llevado a emocionantes nuevas aplicaciones de sistemas de bases de datos. La proliferación de sitios web de redes sociales, como Facebook, Twitter y Flickr, entre muchos otros, ha requerido la creación ación de enormes bases de datos que almacenan datos no tradicionales, como publicaciones, tweets, imágenes y videoclips. Nuevos tipos de sistemas de bases de datos, a menudo denominados **big data** Los sistemas de almacenamiento, o **sistemas NOSQL** , se han creado para administrar datos para **redes** sociales. aplicaciones de medios. Este tipo de sistemas también son utilizados por empresas como Google, Amazon y Yahoo, para administrar los datos requeridos en su búsqueda web motores, así como para proporcionar **almacenamiento en la nube** , mediante el cual se proporciona a los usuarios capacidades de envejecimiento en la Web para administrar todo tipo de datos, incluidos documentos, programas, imágenes, videos y correos electrónicos. Daremos una descripción general de estos nuevos tipos. de los sistemas de bases de datos en el Capítulo 24.

Ahora mencionamos algunas otras aplicaciones de bases de datos. La amplia disponibilidad de
La tecnología de fotografía y video en teléfonos móviles y otros dispositivos ha hecho posible

3

Página 4

4 Capítulo 1 Bases de datos y usuarios de bases de datos

almacenar imágenes, clips de audio y secuencias de video digitalmente. Estos tipos de archivos se
un componente importante de **las bases** de **datos multimedia** . **Información geográfica**
Los sistemas (GIS) pueden almacenar y analizar mapas, datos meteorológicos e imágenes de satélite.
Los almacenes de datos y **los** sistemas de **procesamiento analítico en línea (OLAP)** se utilizan en
muchas empresas para extraer y analizar información empresarial útil de muy grandes
bases de datos para apoyar la toma de decisiones. **Tecnología de base de datos activa** y en **tiempo real**
se utiliza para controlar procesos industriales y de fabricación. Y **búsqueda de** base de datos
se están aplicando **técnicas** a la World Wide Web para mejorar la búsqueda de
información que necesitan los usuarios que navegan por Internet.

Sin embargo, para comprender los fundamentos de la tecnología de bases de datos, debemos comenzar
desde los conceptos básicos de las aplicaciones de bases de datos tradicionales. En la sección 1.1 comenzamos por definir
ing una base de datos, y luego explicamos otros términos básicos. En la Sección 1.2, proporcionamos una
ejemplo simple de base de datos de la UNIVERSIDAD para ilustrar nuestra discusión. Sección 1.3
describe algunas de las principales características de los sistemas de bases de datos, y las Secciones 1.4
y 1.5 categorizar los tipos de personal cuyos trabajos implican el uso e interacción
con sistemas de bases de datos. Las secciones 1.6, 1.7 y 1.8 ofrecen una discusión más detallada
de las diversas capacidades proporcionadas por los sistemas de bases de datos y discutir algunas
aplicaciones de bases de datos. La sección 1.9 resume el capítulo.

El lector que desee una rápida introducción a los sistemas de bases de datos puede estudiar
Secciones 1.1 a 1.5, luego salte o navegue por las Secciones 1.6 a 1.8 y
continúe con el Capítulo 2.

1.1 Introducción

Las bases de datos y la tecnología de bases de datos han tenido un impacto importante en el uso creciente de
ordenadores. Es justo decir que las bases de datos juegan un papel fundamental en casi todas las áreas donde
se utilizan computadoras, incluidos negocios, comercio electrónico, redes sociales, ingeniería
neering, medicina, genética, derecho, educación y bibliotecología. La base de datos de palabras
se usa tan comúnmente que debemos comenzar por definir qué es una base de datos. Nuestra inicial
la definición es bastante general.

Una **base de datos** es una colección de datos relacionados. [1] Por **datos** , nos referimos a hechos conocidos que pueden
ser registrados y que tengan un significado implícito. Por ejemplo, considere los nombres,
números de teléfono y direcciones de las personas que conoce. Hoy en día, estos datos son
normalmente se almacenan en teléfonos móviles, que tienen su propio software de base de datos simple.
Estos datos también pueden registrarse en una libreta de direcciones indexada o almacenarse en un disco duro.
drive, usando una computadora personal y software como Microsoft Access o Excel.
Esta colección de datos relacionados con un significado implícito es una base de datos.

La definición anterior de base de datos es bastante general; por ejemplo, podemos considerar

la recopilación de palabras que componen esta página de texto para ser datos relacionados y por lo tanto para

[1] Usaremos la palabra datos tanto en singular como en plural, como es común en la literatura sobre bases de datos; el contexto determinará si es singular o plural. En inglés estándar, los datos se utilizan para plural y datum para singular.

Página 5

constituyen una base de datos. Sin embargo, el uso común del término base de datos suele ser más restringido. Una base de datos tiene las siguientes propiedades implícitas:

- Una base de datos representa algún aspecto del mundo real, a veces llamado **minimundo** o el **universo del discurso (UoD)** . Cambios en el mini mundo se reflejan en la base de datos.
- Una base de datos es una colección de datos lógicamente coherente con algunos sentido. Una variedad aleatoria de datos no se puede denominar correctamente base de datos.
- Una base de datos está diseñada, construida y poblada con datos para un propósito específico. Tiene un grupo previsto de usuarios y algunas aplicaciones preconcebidas en que estos usuarios están interesados.

En otras palabras, una base de datos tiene alguna fuente de la que se derivan los datos, en cierto grado de interacción con eventos en el mundo real, y una audiencia que está activamente interactuando ested en su contenido. Los usuarios finales de una base de datos pueden realizar transacciones comerciales (por ejemplo, un cliente compra una cámara) o pueden ocurrir eventos (por ejemplo, un empleado tiene un bebé) que provocan cambios en la información de la base de datos. En orden Para que una base de datos sea precisa y confiable en todo momento, debe ser un fiel reflejo de el mini mundo que representa; por lo tanto, los cambios deben reflejarse en los datos base lo antes posible.

Una base de datos puede ser de cualquier tamaño y complejidad. Por ejemplo, la lista de nombres y Las direcciones mencionadas anteriormente pueden consistir en solo unos pocos cientos de registros, cada uno con un estructura simple. Por otro lado, el catálogo informatizado de una gran biblioteca puede contener medio millón de entradas organizadas en diferentes categorías, por apellido de la autora de María, por tema, por título del libro, con cada categoría organizada alfabéticamente. Se mantendría una base de datos de mayor tamaño y complejidad. por una empresa de redes sociales como Facebook, que tiene más de mil millones de usuarios. La base de datos debe mantener información sobre qué usuarios están relacionados entre sí. como amigos, las publicaciones de cada usuario, qué usuarios pueden ver cada publicación, y una gran cantidad de otros tipos de información necesaria para el correcto funcionamiento de su página web. Para tales sitios web, se necesita una gran cantidad de bases de datos para mantener seguimiento de la información en constante cambio requerida por el sitio web de redes sociales.

Un ejemplo de una gran base de datos comercial es Amazon.com. Contiene datos para más de 60 millones de usuarios activos y millones de libros, CD, videos, DVD, juegos, electrónica, indumentaria y otros artículos. La base de datos ocupa más de 42 terabytes (un terabyte equivale a 10^{12} bytes de almacenamiento) y se almacena en cientos de computadoras (llamados servidores). Millones de visitantes acceden a Amazon.com cada día y utilizan base de datos para realizar compras. La base de datos se actualiza continuamente como nuevos libros.

y otros artículos se agregan al inventario, y las cantidades de existencias se actualizan como las compras se tramitan.

Una base de datos puede generarse y mantenerse manualmente o puede ser ized. Por ejemplo, un catálogo de tarjetas de biblioteca es una base de datos que se puede crear y mantenido manualmente. Se puede crear y mantener una base de datos computarizada ya sea por un grupo de programas de aplicación escritos específicamente para esa tarea o por un

Página 6

sistema de administración de base de datos. Por supuesto, solo nos preocupa la informática bases de datos izadas en este texto.

Un **sistema de administración de bases de datos (DBMS)** es un sistema computarizado que permite usuarios para crear y mantener una base de datos. El DBMS es un software de propósito general sistema que facilita los procesos de definición, construcción, manipulación y compartir bases de datos entre varios usuarios y aplicaciones. **Definiendo** una base de datos implica especificar los tipos de datos, estructuras y restricciones de los datos que se almacenado en la base de datos. La definición de la base de datos o la información descriptiva también almacenado por el DBMS en forma de catálogo de base de datos o diccionario; se llama **metadatos** . **La construcción de** la base de datos es el proceso de almacenar los datos en algunos medio de almacenamiento controlado por el DBMS. **Manipular** una base de datos incluye funciones como consultar la base de datos para recuperar datos específicos, actualizar los datos base para reflejar los cambios en el mini mundo y generar informes a partir de los datos. **Compartir** una base de datos permite que varios usuarios y programas accedan a la base de datos. simultaneamente.

Un **programa de aplicación** accede a la base de datos enviando consultas o solicitudes de datos al DBMS. Una **consulta** [2] normalmente hace que se recuperen algunos datos; una **transacción** puede hacer que se lean algunos datos y que se escriban algunos datos en la base de datos.

Otras funciones importantes proporcionadas por el DBMS incluyen proteger la base de datos y mantenerlo durante un largo período de tiempo. **La protección** incluye protección del sistema ción contra el mal funcionamiento del hardware o software (o bloqueos) y la protección de seguridad contra el acceso no autorizado o malintencionado. Una base de datos grande típica puede tener una vida ciclo de muchos años, por lo que el DBMS debe poder **mantener** el sistema de base de datos permitiendo que el sistema evolucione a medida que los requisitos cambian con el tiempo.

No es absolutamente necesario utilizar software DBMS de propósito general para implementar una base de datos informatizada. Es posible escribir un conjunto personalizado de programas para crear Comió y mantuvo la base de datos, de hecho creando un software DBMS de propósito especial para una aplicación específica, como reservas de aerolíneas. En cualquier caso, ya sea que utilizar un DBMS de propósito general o no: una cantidad considerable de software complejo está desplegado. De hecho, la mayoría de los DBMS son sistemas de software muy complejos.

Para completar nuestras definiciones iniciales, llamaremos a la base de datos y al software DBMS juntos un **sistema de base de datos** . La figura 1.1 ilustra algunos de los conceptos que tenemos discutido hasta ahora.

1.2 Un ejemplo

Consideremos un ejemplo simple con el que la mayoría de los lectores pueden estar familiarizados: a Base de datos de la UNIVERSIDAD para mantener información sobre estudiantes, cursos, y calificaciones en un entorno universitario. La figura 1.2 muestra la estructura de la base de datos y algunos registros de datos de muestra. La base de datos está organizada en cinco archivos, cada uno de los cuales

[2] El término consulta, que originalmente significaba una pregunta o una indagación, a veces se usa de manera vaga para todo tipo de interacciones con bases de datos, incluida la modificación de los datos.

Página 7

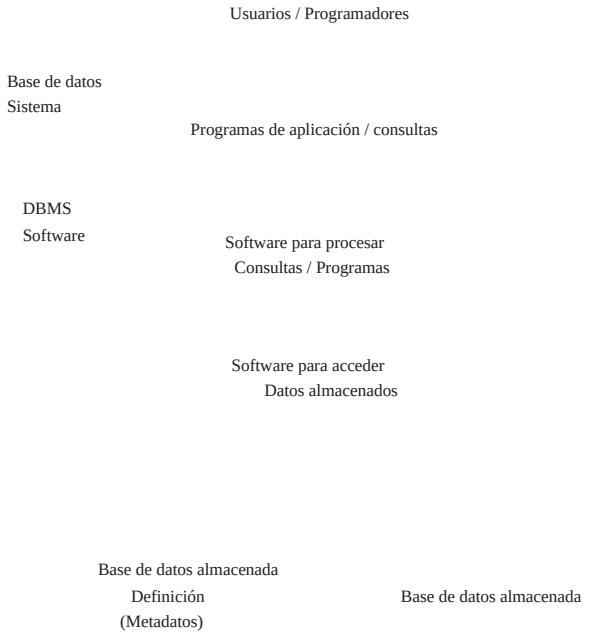


Figura 1.1
Una base de datos simplificada entorno del sistema.

almacena **registros** de **datos** del mismo tipo. [3] El archivo ESTUDIANTE almacena datos sobre cada dent, el archivo COURSE almacena datos en cada curso, el archivo SECTION almacena datos en cada sección de un curso, el archivo GRADE_REPORT almacena las calificaciones que los estudiantes recibir en las distintas secciones que han completado, y el archivo PRERREQUISITO almacena los requisitos previos de cada curso.

Para definir esta base de datos, debemos especificar la estructura de los registros de cada archivo por especificando los diferentes tipos de **elementos** de **datos** que se almacenarán en cada registro. En Figura 1.2, cada registro de ESTUDIANTE incluye datos para representar el Nombre del estudiante , Student_number , Class (como estudiante de primer año o '1', estudiante de segundo año o '2', etc.),

y Mayor (como matemáticas o 'MATH' e informática o 'CS'); cada
El registro de CURSO incluye datos para representar el nombre de curso, número de curso ,
Credit_hours y Department (el departamento que ofrece el curso), y así
en. También debemos especificar un **tipo de datos** para cada elemento de datos dentro de un registro. por
Por ejemplo, podemos especificar que el Nombre del ESTUDIANTE es una cadena de caracteres alfabéticos,
Student_number of STUDENT es un número entero y Grade of GRADE_REPORT es un

3 Aquí usamos el término archivo de manera informal. A nivel conceptual, un archivo es una colección de registros que pueden
no se puede pedir.

Página 8

ESTUDIANTE

Nombre	Número de estudiante	Clase	Mayor
Herrero	17	1	CS
marrón	8	2	CS

CURSO

Nombre del curso	Número de curso	Departamento de horas de crédito	
Introducción a la informática	CS1310	4	CS
Estructuras de datos	CS3320	4	CS
Matemáticas discretas	MATH2410	3	MATEMÁTICAS
Base de datos	CS3380	3	CS

SECCIÓN

Identificador de sección	Número de curso	Semestre	Año	Instructor
85	MATH2410	Otoño	07	Rey
92	CS1310	Otoño	07	Anderson
102	CS3320	Primavera	08	Knuth
112	MATH2410	Otoño	08	Chang
119	CS1310	Otoño	08	Anderson
135	CS3380	Otoño	08	Roca

GRADE_REPORT

Número de estudiante	Section_identifier	Grado
17	112	si
17	119	C
8	85	UNA
8	92	UNA
8	102	si

8 135 UNA

REQUISITO PREVIO

Numero de curso	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figura 1.2
Una base de datos que almacena estudiante y curso información.

Página 9

carácter único del conjunto {'A', 'B', 'C', 'D', 'F', 'I'}. También podemos usar una codificación esquema para representar los valores de un elemento de datos. Por ejemplo, en la Figura 1.2 representamos resiente la Clase de un ESTUDIANTE como 1 para el primer año, 2 para el segundo, 3 para el tercer año, 4 para estudiantes de último año y 5 para estudiantes de posgrado.

Para construir la base de datos de la UNIVERSIDAD , almacenamos datos para representar a cada estudiante, curso, sección, informe de calificaciones y prerrequisito como un registro en el archivo correspondiente. Tenga en cuenta que los registros de los distintos archivos pueden estar relacionados. Por ejemplo, el récord de Smith en el archivo ESTUDIANTE está relacionado con dos registros en el archivo GRADE_REPORT que especifique las calificaciones de Smith en dos secciones. De manera similar, cada registro en el PRERREQUISITO archivo relaciona dos registros del curso: uno representa el curso y el otro representa ing el prerrequisito. La mayoría de las bases de datos de tamaño mediano y grande incluyen muchos tipos de registros y tienen muchas relaciones entre los registros.

La manipulación de la base de datos implica consultas y actualizaciones. Ejemplos de consultas son como sigue:

- Recuperar el expediente académico (una lista de todos los cursos y calificaciones) de 'Smith'
- Enumere los nombres de los estudiantes que tomaron la sección del curso 'Base de datos' ofrecidos en el otoño de 2008 y sus calificaciones en esa sección
- Enumere los requisitos previos del curso 'Base de datos'

Entre los ejemplos de actualizaciones se incluyen los siguientes:

- Cambiar la clase de 'Smith' a segundo año
- Crear una nueva sección para el curso 'Base de datos' para este semestre
- Ingrese una calificación de 'A' para 'Smith' en la sección 'Base de datos' del último semestre

Estas consultas y actualizaciones informales deben especificarse con precisión en el lenguaje de consulta. calibre del DBMS antes de que puedan ser procesados.

En esta etapa, es útil describir la base de datos como parte de una empresa mayor conocido como un sistema de información dentro de una organización. La tecnología de la información departamento de nología (TI) dentro de una organización diseña y mantiene una información sistema que consta de varias computadoras, sistemas de almacenamiento, software de aplicación, y bases de datos. Diseño de una nueva aplicación para una base de datos existente o diseño de una

La nueva base de datos comienza con una fase llamada **especificación de requisitos y análisis** . Estos requisitos se documentan en detalle y se transforman en un **diseño conceptual** que puede ser representado y manipulado usando alguna computadora herramientas erizadas para que se pueda mantener, modificar y transformar fácilmente en un implementación de la base de datos. (Introduciremos un modelo llamado Entidad-Relación-modelo de barco en el Capítulo 3 que se utiliza para este propósito.) El diseño luego se traduce a un **diseño lógico** que puede expresarse en un modelo de datos implementado en un DBMS mercial. (Se discuten varios tipos de DBMS a lo largo del texto, con una énfasis en los DBMS relacionales en los Capítulos 5 al 9.)

La etapa final es **el diseño físico** , durante el cual se proporcionan especificaciones adicionales para almacenar y acceder a la base de datos. El diseño de la base de datos está implementado, poblado con datos reales y mantenidos continuamente para reflejar el estado del mini mundo.

Página 10

1.3 Características del enfoque de base de datos

Varias características distinguen el enfoque de la base de datos del enfoque más antiguo de escribir programas personalizados para acceder a los datos almacenados en archivos. En **procesamiento de archivos** tradicional , cada usuario define e implementa los archivos necesarios para un aplicación de software específica como parte de la programación de la aplicación. Por ejemplo, un usuario, la oficina de informes de calificaciones, puede mantener archivos sobre los estudiantes y sus calificaciones. Se implementan programas para imprimir el expediente académico de un estudiante e ingresar nuevas calificaciones como parte de la aplicación. Un segundo usuario, la oficina de contabilidad, puede realizar un seguimiento de cuotas de los estudiantes y sus pagos. Aunque ambos usuarios están interesados en datos sobre estudiantes, cada usuario mantiene archivos y programas separados para manipularlos archivos, porque cada uno requiere algunos datos que no están disponibles en los archivos del otro usuario. Esta redundancia en la definición y el almacenamiento de datos da como resultado un desperdicio de espacio de almacenamiento y en esfuerzos redundantes para mantener datos comunes actualizados.

En el enfoque de la base de datos, un único repositorio mantiene los datos que se definen una vez y luego varios usuarios acceden repetidamente a través de consultas, transacciones y programas de aplicación. Las principales características del enfoque de base de datos frente al El enfoque de procesamiento de archivos es el siguiente:

- Naturaleza autodescriptiva de un sistema de base de datos
- Aislamiento entre programas y datos, y abstracción de datos
- Soporte de múltiples vistas de los datos
- Intercambio de datos y procesamiento de transacciones multiusuario

Describimos cada una de estas características en una sección separada. Discutiremos adicionales características de los sistemas de bases de datos en las Secciones 1.6 a 1.8.

1.3.1 Naturaleza autodescriptiva de un sistema de base de datos

Una característica fundamental del enfoque de base de datos es que el sistema de base de datos contiene no solo la base de datos en sí, sino también una definición o descripción completa de

la estructura y las limitaciones de la base de datos. Esta definición se almacena en la catalog, que contiene información como la estructura de cada archivo, el tipo y el almacenamiento formato de edad de cada elemento de datos y varias restricciones sobre los datos. La información almacenados en el catlogo se llaman **metadatos** , y describen la estructura del pribase de datos de Mary (Figura 1.1). Es importante señalar que algunos tipos más nuevos de datos: Los sistemas base, conocidos como sistemas NOSQL, no requieren metadatos. Más bien los datos se almacena como **datos autodescriptivos** que incluyen los nombres y valores de los elementos de datos juntos en una estructura (ver Capítulo 24).

El catálogo es utilizado por el software DBMS y también por los usuarios de la base de datos que necesitan información sobre la estructura de la base de datos. Un software DBMS de propósito general El paquete no está escrito para una aplicación de base de datos específica. Por tanto, debe referirse al catálogo para conocer la estructura de los archivos en una base de datos específica, como el tipo y formato de los datos a los que accederá. El software DBMS debe funcionar igualmente bien con cualquier número de aplicaciones de base de datos, por ejemplo, una base de datos universitaria, una

Página 11

base de datos bancaria, o una base de datos de la empresa, siempre que la definición de la base de datos sea almacenados en el catálogo.

En el procesamiento de archivos tradicional, la definición de datos suele ser parte del proceso de aplicación. gramos ellos mismos. Por lo tanto, estos programas están limitados a trabajar con un solo base de datos específica, cuya estructura se declara en los programas de aplicación. por Por ejemplo, un programa de aplicación escrito en C ++ puede tener una estructura o una declaración de clase. ciones. Mientras que el software de procesamiento de archivos solo puede acceder a bases de datos específicas, DBMS El software puede acceder a diversas bases de datos extrayendo las definiciones de la base de datos de el catálogo y utilizando estas definiciones.

Para el ejemplo que se muestra en la Figura 1.2, el catálogo DBMS almacenará las definiciones de todos los archivos mostrados. La figura 1.3 muestra algunas entradas en un catálogo de base de datos. Siempre que un se hace una solicitud para acceder, digamos, el nombre de un registro de ESTUDIANTE , el software DBMS se refiere al catálogo para determinar la estructura del expediente del ESTUDIANTE y la posición y tamaño del elemento de datos de Nombre dentro de un registro de ESTUDIANTE . Por el contrario, en un típico aplicación de procesamiento de archivos, la estructura del archivo y, en el caso extremo, la la ubicación del nombre dentro de un registro de ESTUDIANTE ya está codificada dentro de cada programa que accede a este elemento de datos.

RELACIONES

Relation_name	No_de_columnas
ESTUDIANTE	4
CURSO	4
SECCIÓN	5
GRADE_REPORT	3
REQUISITO PREVIO	2

COLUMNAS

Figura 1.3
Un ejemplo de catálogo de base de datos para la base de datos en Figura 1.2.

Column_name	Tipo de datos	Pertenece_a_relación
Nombre	Personaje (30)	ESTUDIANTE
Número de estudiante	Personaje (4)	ESTUDIANTE
Clase	Entero (1)	ESTUDIANTE
Mayor	Major_type	ESTUDIANTE
Nombre del curso	Personaje (10)	CURSO
Numero de curso	XXXXNNNN	CURSO
….	….	… ..
….	….	… ..
….	….	… ..
Prerequisite_number	XXXXNNNN	REQUISITO PREVIO

Nota: Major_type se define como un tipo enumerado con todas las especialidades conocidas.
XXXXNNNN se utiliza para definir un tipo con cuatro caracteres alfabéticos seguidos de cuatro dígitos numéricos.

Pagina 12

1.3.2 Aislamiento entre programas y datos, y abstracción de datos

En el procesamiento de archivos tradicional, la estructura de los archivos de datos está incrustada en la aplicación. programas, por lo que cualquier cambio en la estructura de un archivo puede requerir cambiar todos programas que acceden a ese archivo. Por el contrario, los programas de acceso a DBMS no requieren tales cambios en la mayoría de los casos. La estructura de los archivos de datos se almacena en el catálogo DBMS. log por separado de los programas de acceso. Llamamos a esta propiedad **datos de programa Independencia** .

Por ejemplo, un programa de acceso a archivos puede estar escrito de tal manera que pueda acceder sólo registros de ESTUDIANTES de la estructura que se muestra en la Figura 1.4. Si queremos agregar otro dato para cada registro de ESTUDIANTE , digamos la Fecha de nacimiento , tal programa ya no funcionará y debe cambiarse. Por el contrario, en un entorno DBMS, solo es necesario cambiar la descripción de los registros de ESTUDIANTE en el catálogo (Figura 1.3) para reflejar la inclusión del nuevo elemento de datos Birth_date ; no se cambian los programas. La próxima vez que un programa DBMS se refiere al catálogo, la nueva estructura de Se accederá y utilizará los registros del ESTUDIANTE .

En algunos tipos de sistemas de bases de datos, como los orientados a objetos y relacionales a objetos sistemas (ver el Capítulo 12), los usuarios pueden definir operaciones sobre datos como parte de la base de datos definiciones. Una **operación** (también llamada función o método) se especifica en dos partes. La interfaz (o firma) de una operación incluye el nombre de la operación y los tipos de datos de sus argumentos (o parámetros). La implementación (o método) de la operación se especifica por separado y se puede cambiar sin afectar la inter- cara. Los programas de aplicación de usuario pueden operar sobre los datos invocando estas operaciones. ciones a través de sus nombres y argumentos, independientemente de cómo se realicen las operaciones implementado. Esto puede denominarse **independencia del funcionamiento del programa** .

La característica que permite la independencia de los datos del programa y el funcionamiento del programa. la independencia se llama **abstracción de datos** . Un DBMS proporciona a los usuarios un **concepto**

Representación de datos que no incluye muchos de los detalles de cómo son los datos.
almacenados o cómo se implementan las operaciones. De manera informal, un **modelo de datos** es un tipo de
abstracción de datos que se utiliza para proporcionar esta representación conceptual. Los datos
El modelo utiliza conceptos lógicos, como objetos, sus propiedades y sus interrelaciones.
asociaciones, que pueden ser más fáciles de entender para la mayoría de los usuarios que el almacenamiento de la computadora
conceptos. Por tanto, el modelo de datos oculta los detalles de almacenamiento e implementación que son
no es de interés para la mayoría de los usuarios de bases de datos.

Al observar el ejemplo de las Figuras 1.2 y 1.3, la implementación interna del
El archivo ESTUDIANTE puede definirse por su longitud de registro: el número de caracteres
(bytes) en cada registro, y cada elemento de datos puede especificarse por su byte inicial
dentro de un registro y su longitud en bytes. El expediente del ESTUDIANTE quedaría así representado
enviado como se muestra en la Figura 1.4. Pero un usuario típico de una base de datos no se preocupa por la
ubicación de cada elemento de datos dentro de un registro o su longitud; más bien, el usuario está preocupado
que cuando se hace una referencia al Nombre del ESTUDIANTE , se devuelve el valor correcto.
En la Figura 1.2 se muestra una representación conceptual de los registros del ESTUDIANTE . Muchos
otros detalles de la organización del almacenamiento de archivos, como las rutas de acceso especificadas en un

Página 13

Nombre del elemento de datos	Posición inicial en el registro	Longitud en caracteres (bytes)
Nombre	1	30
Número de estudiante	31	4
Clase	35	1
Mayor	36	4

Figura 1.4
Formato de almacenamiento interno para un registro de ESTUDIANTE, basado en la base de datos catálogo en la Figura 1.3.

archivo: el DBMS puede ocultarlo a los usuarios de la base de datos; discutimos los detalles de almacenamiento en Capítulos 16 y 17.

En el enfoque de la base de datos, la estructura y organización detalladas de cada archivo son
almacenados en el catálogo. Los usuarios de la base de datos y los programas de aplicación se refieren al concepto
representación real de los archivos, y el DBMS extrae los detalles del almacenamiento de archivos
del catálogo cuando sean necesarios para los módulos de acceso a archivos DBMS. Muchos
Se pueden utilizar modelos de datos para proporcionar esta abstracción de datos a los usuarios de la base de datos. La mayor
parte de este texto está dedicada a presentar varios modelos de datos y los conceptos que
utilizar para abstraer la representación de datos.

En las bases de datos relacionales y orientadas a objetos, el proceso de abstracción incluye
no solo la estructura de los datos, sino también las operaciones sobre los datos. Estas operaciones
proporcionar una abstracción de las actividades del mini mundo comúnmente comprendidas por los usuarios.
Por ejemplo, una operación CALCULATE_GPA se puede aplicar a un objeto ESTUDIANTE
para calcular el promedio de calificaciones. Estas operaciones pueden ser invocadas por el usuario.
consultas o programas de aplicación sin tener que conocer los detalles de cómo
se implementan las operaciones.

1.3.3 Soporte de múltiples vistas de los datos

Una base de datos suele tener muchos tipos de usuarios, cada uno de los cuales puede requerir un perspectiva o **vista** de la base de datos. Una vista puede ser un subconjunto de la base de datos o puede contienen **datos virtuales** que se derivan de los archivos de la base de datos pero que no se almacenan explícitamente. Es posible que algunos usuarios no necesiten saber si los datos a los que hacen referencia están almacenados o derivado. Un DBMS multiusuario cuyos usuarios tienen una variedad de aplicaciones distintas debe proporcionar facilidades para definir múltiples vistas. Por ejemplo, un usuario de la base de datos de la Figura 1.2 puede estar interesado solo en acceder e imprimir la transcripción de cada estudiante; la vista para este usuario se muestra en la Figura 1.5 (a). Un segundo usuario, que está interesado Solo se comprueba que los alumnos hayan cumplido con todos los requisitos previos de cada curso. para lo cual el estudiante se registra, puede requerir la vista que se muestra en la Figura 1.5 (b).

1.3.4 Intercambio de datos y procesamiento de transacciones multiusuario

Un DBMS multiusuario, como su nombre lo indica, debe permitir que varios usuarios accedan al base de datos al mismo tiempo. Esto es esencial si los datos para múltiples aplicaciones integrado y mantenido en una sola base de datos. El DBMS debe incluir **concurrencia** software de **control** para garantizar que varios usuarios intenten actualizar los mismos datos

Página 14

(una)

TRANSCRIPCIÓN

Nombre del estudiante	Student_transcript				
	Numero de curso	Grado	Semestre	Año	Section_id
Herrero	CS1310	C	Otoño	08	119
	MATH2410	si	Otoño	08	112
marrón	MATH2410	UNA	Otoño	07	85
	CS1310	UNA	Otoño	07	92
	CS3320	si	Primavera	08	102
	CS3380	UNA	Otoño	08	135

(si)

COURSE_PREREQUISITES

Nombre del curso	Numero de curso	Prerrequisitos
Base de datos	CS3380	CS3320
		MATH2410
Estructuras de datos	CS3320	CS1310

Figura 1.5
Dos vistas derivadas de la base de datos de la Figura 1.2. (a) La vista TRANSCRIPCIÓN.
(b) La vista COURSE_PREREQUISITES.

Hágalo de forma controlada para que el resultado de las actualizaciones sea correcto. Para examen- Por ejemplo, cuando varios agentes de reservas intentan asignar un asiento en un vuelo de aerolínea,

El DBMS debe garantizar que solo un agente pueda acceder a cada puesto a la vez para asignación a un pasajero. Estos tipos de aplicaciones generalmente se denominan en **línea. aplicaciones de procesamiento de transacciones (OLTP)** . Un papel fundamental del multiusuario El software DBMS es para asegurar que las transacciones concurrentes operen correctamente y eficientemente.

El concepto de **transacción se** ha vuelto fundamental para muchas aplicaciones de bases de datos. UNA transacción es un programa o proceso en ejecución que incluye una o más bases de datos accesos, como lectura o actualización de registros de bases de datos. Cada transacción es compatible planteado para ejecutar un acceso a la base de datos lógicamente correcto si se ejecuta en su totalidad sin- interferencia de otras transacciones. El DBMS debe hacer cumplir varios propiedades de la transacción. La propiedad de **aislamiento** asegura que cada transacción parece ejecutarse de forma aislada de otras transacciones, a pesar de que cientos de las transacciones pueden estar ejecutándose al mismo tiempo. La propiedad de **atomicidad** asegura que o se ejecutan todas las operaciones de la base de datos en una transacción o ninguna. Dis- Maldecir las transacciones en detalle en la Parte 9.

Las características anteriores son importantes para distinguir un DBMS de los tradicionales. software de procesamiento de archivos nacional. En la Sección 1.6 discutimos características adicionales que caracterizar un DBMS. Sin embargo, primero clasificamos los diferentes tipos de personas que trabajan en un entorno de sistema de base de datos.

Página 15

1.4 Actores en escena

Para una pequeña base de datos personal, como la lista de direcciones discutida en la Sección 1.1, una persona normalmente define, construye y manipula la base de datos, y hay no compartir. Sin embargo, en las grandes organizaciones, muchas personas están involucradas diseño, uso y mantenimiento de una gran base de datos con cientos o miles de usuarios. En esta sección identificamos a las personas cuyos trabajos implican el uso diario de una gran base de datos; los llamamos los actores en escena. En la Sección 1.5 consideramos personas a las que se les puede llamar trabajadores detrás de escena, aquellos que trabajan para mantener el entorno del sistema de base de datos, pero que no están activamente interesados en los datos contenidos básicos como parte de su trabajo diario.

1.4.1 Administradores de bases de datos

En cualquier organización donde muchas personas usan los mismos recursos, es necesario un administrador jefe para supervisar y administrar estos recursos. En un entorno de base de datos , el recurso principal es la base de datos en sí, y el recurso secundario es el DBMS y software relacionado. La administración de estos recursos es responsabilidad de el **administrador de** la **base de datos (DBA)** . El DBA es responsable de autorizar el acceso a la base de datos, coordinando y monitoreando su uso, y adquiriendo software y recursos de hardware según sea necesario. El DBA es responsable de problemas como la seguridad brechas de seguridad y tiempo de respuesta deficiente del sistema. En organizaciones grandes, el DBA es asistido por un personal que realiza estas funciones.

1.4.2 Diseñadores de bases de datos

Los diseñadores de bases de datos son responsables de identificar los datos que se almacenarán en los datos-
base y para elegir las estructuras adecuadas para representar y almacenar estos datos. Estas
La mayoría de las tareas se realizan antes de que la base de datos se implemente y la población
lated con datos. Es responsabilidad de los diseñadores de bases de datos comunicarse con
todos los posibles usuarios de la base de datos para comprender sus requisitos y crear
comió un diseño que cumpla con estos requisitos. En muchos casos, los diseñadores están en el
personal del DBA y se le pueden asignar otras responsabilidades de personal después de la base de datos
El diseño está terminado. Los diseñadores de bases de datos suelen interactuar con cada grupo potencial
de los usuarios y desarrollar **vistas** de la base de datos que cumplan con los datos y el procesamiento
requisitos de estos grupos. Luego, cada vista se analiza e integra con el
vistas de otros grupos de usuarios. El diseño final de la base de datos debe ser capaz de soportar
los requisitos de todos los grupos de usuarios.

1.4.3 Usuarios finales

Los usuarios finales son las personas cuyos trabajos requieren acceso a la base de datos para realizar consultas,
actualizar y generar informes; la base de datos existe principalmente para su uso. Ahí
Hay varias categorías de usuarios finales:

- **Los usuarios finales** ocasionales acceden ocasionalmente a la base de datos, pero pueden necesitar información ent cada vez. Utilizan una interfaz de consulta de base de datos sofisticada

Página 16

para especificar sus solicitudes y suelen ser gerentes de nivel medio o alto o
otros navegadores ocasionales.

- **Los usuarios finales ingenuos** o **paramétricos** constituyen una parte considerable de la base de datos usuarios finales. Su principal función laboral gira en torno a la consulta constante
y actualizar la base de datos, utilizando tipos estándar de consultas y actualizaciones—
llamadas **transacciones enlatadas,** que han sido cuidadosamente programadas y
probado. Muchas de estas tareas ahora están disponibles como **aplicaciones móviles** para su uso con
dispositivos móviles. Las tareas que realizan estos usuarios son variadas. Unos pocos
ejemplos son:

  Los clientes bancarios y los cajeros controlan los saldos de las cuentas y registran los retiros.
  y depósitos.

  Agentes de reservas o clientes de aerolíneas, hoteles y compañías de alquiler de automóviles.
  Los panies verifican la disponibilidad para una solicitud determinada y hacen reservas.

  Los empleados de las estaciones receptoras de las empresas de transporte ingresan al paquete
  identificaciones mediante códigos de barras e información descriptiva mediante botones
  actualizar una base de datos central de paquetes recibidos y en tránsito.

  Los usuarios de redes sociales publican y leen elementos en sitios web de redes sociales.

- Los **usuarios finales sofisticados** incluyen ingenieros, científicos, analistas de negocios y otros que se familiaricen completamente con las instalaciones del DBMS
para implementar sus propias aplicaciones para satisfacer sus complejos requisitos
mentos.

- **Los usuarios independientes** mantienen bases de datos personales mediante el uso de programas paquetes de gran que proporcionan menús o gráficos fáciles de usar interfaces. Un ejemplo es el usuario de un paquete de software financiero que almacena una variedad de datos financieros personales.

Un DBMS típico proporciona múltiples facilidades para acceder a una base de datos. Usuarios finales ingenuos necesita aprender muy poco sobre las facilidades que ofrece el DBMS; simplemente tienen comprender las interfaces de usuario de las aplicaciones móviles o las transacciones estándar diseñado e implementado para su uso. Los usuarios ocasionales aprenden solo algunas instalaciones que pueden usar repetidamente. Los usuarios sofisticados intentan aprender la mayoría de las instalaciones de DBMS para lograr sus complejos requisitos. Los usuarios independientes suelen convertirse muy competente en el uso de un paquete de software específico.

1.4.4 Analistas de sistemas y programadores de aplicaciones (Ingenieros de software)

Los analistas de sistemas determinan los requisitos de los usuarios finales, especialmente los ingenuos y usuarios finales paramétricos y desarrollar especificaciones para transacciones enlatadas estándar que cumplan estos requisitos. **Los programadores de aplicaciones** implementan estos cationes como programas; luego prueban, depuran, documentan y mantienen estos archivos enlatados actas. Dichos analistas y programadores, comúnmente conocidos como **software desarrolladores** o **ingenieros de software,** deben estar familiarizados con la gama completa de capacidades bilidades proporcionadas por el DBMS para realizar sus tareas.

Página 17

1.5 Trabajadores detrás de escena

Además de los que diseñan, utilizan y administran una base de datos, otros están asociados atado con el diseño, desarrollo y operación del software y sistema DBMS ambiente. Por lo general, estas personas no están interesadas en el contenido de la base de datos. sí mismo. Los llamamos los trabajadores detrás de escena, e incluyen a los siguientes categorías:

- Los **diseñadores e implementadores de sistemas DBMS** diseñan e implementan Módulos e interfaces DBMS como paquete de software. Un DBMS es muy sistema de software complejo que consta de muchos componentes o **módulos** , incluyendo módulos para implementar el catálogo, proceso de lenguaje de consulta- procesamiento de interfaz, acceso y almacenamiento en búfer de datos, control de concurrencia rencia y manejo de recuperación y seguridad de datos. El DBMS debe interactuar con otro software del sistema, como el sistema operativo y los compiladores para varios lenguajes de programación.
- **Los desarrolladores de herramientas** diseñan e implementan **herramientas:** los paquetes de software que Facilitar el modelado y diseño de bases de datos, el diseño de sistemas de bases de datos y desempeño mejorado. Las herramientas son paquetes opcionales que a menudo se compran perseguido por separado. Incluyen paquetes para diseño de bases de datos, rendimiento monitorización, lenguaje natural o interfaces gráficas, prototipos, simulación y generación de datos de prueba. En muchos casos, los proveedores de software independientes

desarrollar y comercializar estas herramientas.

- **Operadores y personal de mantenimiento** (personal de administración del sistema) son responsables del funcionamiento real y el mantenimiento del hardware y entorno de software para el sistema de base de datos.

Aunque estas categorías de trabajadores detrás de escena son fundamentales para hacer el sistema de base de datos disponible para los usuarios finales, normalmente no utilizan la base de datos contenidos para sus propios fines.

1.6 Ventajas de utilizar el enfoque DBMS

En esta sección discutimos algunas ventajas adicionales de usar un DBMS y el capacidades que debe poseer un buen DBMS. Estas capacidades son adicionales a las cuatro características principales discutidas en la Sección 1.3. El DBA debe utilizar estos capacidades para lograr una variedad de objetivos relacionados con el diseño, administración y el uso de una gran base de datos multiusuario.

1.6.1 Control de la redundancia

En el desarrollo de software tradicional que utiliza procesamiento de archivos, cada grupo de usuarios mantiene sus propios archivos para manejar sus aplicaciones de procesamiento de datos. Por ejemplo, considere el ejemplo de la base de datos UNIVERSIDAD de la Sección 1.2; aquí, dos grupos de los usuarios pueden ser el personal de registro del curso y la oficina de contabilidad. En el enfoque tradicional, cada grupo mantiene archivos de los estudiantes de forma independiente. los

Página 18

la oficina de contabilidad mantiene datos sobre el registro y la información de facturación relacionada, mientras que la oficina de registro realiza un seguimiento de los cursos y calificaciones de los estudiantes. Otro los grupos pueden duplicar aún más algunos o todos los mismos datos en sus propios archivos.

Esta **redundancia** al almacenar los mismos datos varias veces conduce a varios problemas. Primero, existe la necesidad de realizar una única actualización lógica, como ingresar datos en un nuevo alumno, varias veces: una vez por cada archivo donde se registran los datos del alumno. Esto conduce a la duplicación de esfuerzos. En segundo lugar, el espacio de almacenamiento se desperdicia cuando el mismo los datos se almacenan repetidamente y este problema puede ser grave para grandes bases de datos. En tercer lugar, los archivos que representan los mismos datos pueden volverse inconsistentes. Esto puede pasar porque se aplica una actualización a algunos de los archivos pero no a otros. Incluso si un actualización, como agregar un nuevo alumno, se aplica a todos los archivos apropiados, la Los datos relacionados con el estudiante pueden seguir siendo inconsistentes porque se aplican las actualizaciones. independientemente por cada grupo de usuarios. Por ejemplo, un grupo de usuarios puede ingresar un fecha de nacimiento de dent erróneamente como 'JAN-19-1988', mientras que los otros grupos de usuarios pueden ingrese el valor correcto de 'JAN-29-1988'.

En el enfoque de base de datos, las vistas de diferentes grupos de usuarios se integran durante diseño de base de datos. Idealmente, deberíamos tener un diseño de base de datos que almacene cada lógica elemento de datos, como el nombre de un estudiante o la fecha de nacimiento, en un solo lugar en los datos base. Esto se conoce como **normalización de datos** y garantiza la coherencia y ahorra

espacio de almacenamiento (la normalización de datos se describe en la Parte 6 del texto). Sin embargo, en la práctica, a veces es necesario utilizar **redundancia controlada** para mejorar el rendimiento de las consultas. Por ejemplo, podemos almacenar Student_name y Course_number de forma redundante en un archivo GRADE_REPORT (Figura 1.6 (a)) porque cada vez que recuperamos un registro GRADE_REPORT , queremos recuperar el estudiante nombre y número de curso junto con el grado, número de estudiante e identificación de sección más ardiente. Al colocar todos los datos juntos, no tenemos que buscar varios archivos para lea estos datos. Esto se conoce como **desnormalización** . En tales casos, el DBMS debe

Figura 1.6
Almacenamiento redundante de Student_name y Course_name en GRADE_REPORT.
(a) Datos consistentes.
(b) inconsistente grabar.

(una)

GRADE_REPORT

Student_number	Student_name	Section_identifier	Course_number	Grado
17	Herrero	112	MATH2410	si
17	Herrero	119	CS1310	C
8	marrón	85	MATH2410	UNA
8	marrón	92	CS1310	UNA
8	marrón	102	CS3320	si
8	marrón	135	CS3380	UNA

(si)

GRADE_REPORT

Student_number	Student_name	Section_identifier	Course_number	Grado
17	marrón	112	MATH2410	si

Página 19

tener la capacidad de controlar esta redundancia a fin de prohibir la inconsistencia cies entre los archivos. Esto se puede hacer comprobando automáticamente que el Student_name – Student_number valores en cualquier registro GRADE_REPORT en la figura- Ure 1.6 (a) coincida con uno de los valores de Nombre-Número de estudiante de un registro de ESTUDIANTE (Figura ure 1.2). Del mismo modo, los valores de Section_identifier – Course_number en GRADE_REPORT se puede comparar con los registros de la SECCIÓN . Estos controles se pueden especificar en el DBMS. durante el diseño de la base de datos y automáticamente aplicado por el DBMS siempre que el El archivo GRADE_REPORT está actualizado. La figura 1.6 (b) muestra un registro GRADE_REPORT que es inconsistente con el archivo ESTUDIANTE en la Figura 1.2; este tipo de error se puede ingresar si no se controla la redundancia. ¿Puedes decir qué parte es inconsistente?

1.6.2 Restricción del acceso no autorizado

Cuando varios usuarios comparten una gran base de datos, es probable que la mayoría de los usuarios no autorizado para acceder a toda la información de la base de datos. Por ejemplo, datos financieros tales como salarios y bonificaciones a menudo se considera confidencial, y solo los Las personas autorizadas pueden acceder a dichos datos. Además, algunos usuarios solo pueden tener permiso para recuperar datos, mientras que otros pueden recuperar y actualizar. Por lo tanto, el tipo de operación de acceso (recuperación o actualización) también debe

trolled. Por lo general, los usuarios o grupos de usuarios reciben números de cuenta protegidos por contraseñas, que pueden utilizar para acceder a la base de datos. Un DBMS debería proporcionar un **subsistema de seguridad y autorización** , que el DBA utiliza para crear cuentas y para especificar restricciones de cuenta. Entonces, el DBMS debería hacer cumplir estas restricciones automáticamente. Observe que podemos aplicar controles similares a la Software DBMS. Por ejemplo, solo el personal del DBA puede tener permitido usar ciertos **software privilegiado** , como el software para crear nuevas cuentas. Similar, Los usuarios paramétricos pueden tener acceso a la base de datos solo a través de aplicaciones definidas o transacciones enlatadas desarrolladas para su uso. Discutimos datos seguridad y autorización de base en el Capítulo 30.

1.6.3 Proporcionar almacenamiento persistente para objetos de programa

Las bases de datos se pueden utilizar para proporcionar **almacenamiento persistente** para los objetos y datos del programa. estructuras. Esta es una de las principales razones de los **sistemas de bases de datos orientados a objetos.** (ver el Capítulo 12). Los lenguajes de programación suelen tener estructuras de datos complejas, como estructuras o definiciones de clases en C ++ o Java. Los valores de las variables del programa u objetos se descartan una vez que finaliza un programa, a menos que el programador explícitamente los almacena en archivos permanentes, lo que a menudo implica convertir estos estructuras en un formato adecuado para el almacenamiento de archivos. Cuando surge la necesidad de leer esto datos una vez más, el programador debe convertir del formato de archivo al programa estructura variable u objeto. Los sistemas de bases de datos orientados a objetos son compatibles con lenguajes de programación como C ++ y Java, y el software DBMS realiza de forma matemática las conversiones necesarias. Por tanto, un objeto complejo en C ++ se puede almacenar de forma permanente en un DBMS orientado a objetos. Se dice que tal objeto ser **persistente** , ya que sobrevive a la finalización de la ejecución del programa y puede luego será recuperado directamente por otro programa.

Página 20

El almacenamiento persistente de objetos de programa y estructuras de datos es una función importante. ción de sistemas de bases de datos. Los sistemas de bases de datos tradicionales a menudo sufrieron llamado **problema de desajuste de impedancia** , ya que las estructuras de datos proporcionadas por el Los DBMS eran incompatibles con las estructuras de datos del lenguaje de programación. Los sistemas de bases de datos orientados a objetos suelen ofrecer **compatibilidad de** estructura de datos con uno o más lenguajes de programación orientados a objetos.

1.6.4 Proporcionar estructuras de almacenamiento y búsqueda Técnicas para un procesamiento de consultas eficiente

Los sistemas de bases de datos deben proporcionar capacidades para ejecutar consultas y actualizaciones. Debido a que la base de datos generalmente se almacena en disco, el DBMS debe proporcionar estructuras de datos especializadas y técnicas de búsqueda para acelerar la búsqueda en disco de los registros deseados. Los archivos auxiliares llamados **índices** se utilizan a menudo para este propósito. Los índices se basan típicamente en estructuras de datos de árbol o estructuras de datos hash que son convenientemente modificado para la búsqueda de disco. Para procesar los registros de la base de datos necesarios mediante una consulta en particular, esos registros deben copiarse del disco a la memoria principal.

Por lo tanto, el DBMS a menudo tiene un módulo de **almacenamiento en búfer** o **caché** que mantiene las partes de la base de datos en los búferes de memoria principal. En general, el sistema operativo responde sible para el almacenamiento en búfer de disco a memoria. Sin embargo, debido a que el almacenamiento en búfer de datos es crucial para rendimiento del DBMS, la mayoría de los DBMS realizan su propio almacenamiento en búfer de datos.

El módulo de **procesamiento y optimización** de consultas del DBMS es responsable de elegir un plan de ejecución de consultas eficiente para cada consulta en función de los estructuras de almacenamiento. La elección de qué índices crear y mantener es parte de diseño y ajuste de la base de datos física, que es una de las responsabilidades del DBA personal. Discutimos el procesamiento y la optimización de consultas en la Parte 8 del texto.

1.6.5 Proporcionar respaldo y recuperación

Un DBMS debe proporcionar instalaciones para recuperarse de fallas de hardware o software. El **subsistema** de **respaldo y recuperación** del DBMS es responsable de la recuperación. por ejemplo, si el sistema informático falla en medio de una transacción de actualización compleja ción, el subsistema de recuperación es responsable de asegurarse de que la base de datos esté restaurado al estado en el que estaba antes de que la transacción comenzara a ejecutarse. Copia de seguridad en disco también es necesario en caso de una falla catastrófica del disco. Hablamos de recuperación y copia de seguridad en el Capítulo 22.

1.6.6 Proporcionar múltiples interfaces de usuario

Debido a que muchos tipos de usuarios con distintos niveles de conocimientos técnicos utilizan base, un DBMS debe proporcionar una variedad de interfaces de usuario. Estos incluyen aplicaciones para usuarios móviles, lenguajes de consulta para usuarios ocasionales, interfaces de lenguaje de programación para programadores de aplicaciones, formularios y códigos de comando para usuarios paramétricos, e interfaces basadas en menús e interfaces de lenguaje natural para usuarios independientes. Tanto las interfaces de estilo de formularios como las interfaces controladas por menús se conocen comúnmente como

Página 21

interfaces gráficas de usuario (GUI) . Muchos lenguajes y entornos especializados existen para especificar GUI. Capacidades para proporcionar interfaces GUI web a un bases de datos, o habilitar una base de datos para la Web, también son bastante comunes.

1.6.7 Representar relaciones complejas entre datos

Una base de datos puede incluir numerosas variedades de datos que están interrelacionados en muchos formas. Considere el ejemplo que se muestra en la Figura 1.2. El récord de 'Brown' en el El archivo ESTUDIANTE está relacionado con cuatro registros en el archivo GRADE_REPORT . Similar, Cada registro de sección está relacionado con un registro de curso y con varios Registros GRADE_REPORT : uno por cada estudiante que completó esa sección. UNA DBMS debe tener la capacidad de representar una variedad de relaciones complejas entre los datos, para definir nuevas relaciones a medida que surgen, y para recuperar y actualice los datos relacionados de manera fácil y eficiente.

1.6.8 Hacer cumplir las restricciones de integridad

La mayoría de las aplicaciones de bases de datos tienen ciertas **restricciones de integridad** que deben mantenerse los datos. Un DBMS debe proporcionar capacidades para definir y hacer cumplir estos limitaciones. El tipo más simple de restricción de integridad implica especificar un dato escriba para cada elemento de datos. Por ejemplo, en la Figura 1.3, especificamos que el valor de el elemento de datos de la clase dentro de cada registro de ESTUDIANTE debe ser un número entero de un dígito y que el valor de Nombre debe ser una cadena de no más de 30 caracteres alfabéticos. Restringir el valor de Class entre 1 y 5 sería una restricción adicional que no se muestra en el catálogo actual. Un tipo de restricción más complejo que ocurre con frecuencia implica especificar que un registro en un archivo debe estar relacionado con registros en otros archivos. Por ejemplo, en la Figura 1.2, podemos especificar que cada sección El registro debe estar relacionado con el registro de un curso. Esto se conoce como **integridad referencial** restricción. Otro tipo de restricción especifica la unicidad de los valores de los elementos de datos, por ejemplo, cada registro de curso debe tener un valor único para Course_number . Esto es conocida como restricción de **clave** o **unicidad** . Estas restricciones se derivan de la significado o **semántica** de los datos y del minimundo que representa. Es el responsabilidad de los diseñadores de la base de datos para identificar las restricciones de integridad durante diseño de base de datos. Algunas restricciones se pueden especificar al DBMS y automáticamente en vigor. Es posible que sea necesario verificar otras restricciones mediante programas de actualización o en el hora de entrada de datos. Para aplicaciones típicas de gran tamaño, es habitual llamar a este tipo de con- restringe **las reglas comerciales** .

Un elemento de datos puede ingresarse erróneamente y aún así cumplir con el requisito de integridad especificado. tensiones. Por ejemplo, si un estudiante recibe una calificación de 'A' pero se ingresa una calificación de 'C' en la base de datos, el DBMS no puede descubrir este error automáticamente porque 'C' es un valor válido para el tipo de datos de calificación . Dichos errores de entrada de datos solo se pueden descubrir manualmente (cuando el alumno recibe la calificación y se queja) y se corrige más tarde actualizando la base de datos. Sin embargo, una calificación de 'Z' se rechazaría automáticamente por el DBMS porque 'Z' no es un valor válido para el tipo de datos de Grado . Cuando dis- Para mal cada modelo de datos en los capítulos siguientes, presentaremos reglas que pertenecen a

Página 22

ese modelo implícitamente. Por ejemplo, en el modelo Entidad-Relación del Capítulo 3, una relación debe involucrar al menos a dos entidades. Reglas que pertenecen a un dato específico modelo se denominan **reglas inherentes** del modelo de datos.

1.6.9 Permitir inferencias y acciones Usar reglas y activadores

Algunos sistemas de bases de datos proporcionan capacidades para definir reglas de deducción para inferir Obtener nueva información de los hechos almacenados en la base de datos. Tales sistemas se llaman **sistemas de base de datos deductivos** . Por ejemplo, puede haber reglas complejas en el aplicación mundial para determinar cuándo un estudiante está en libertad condicional. Estos pueden ser especificadas declarativamente como **reglas** , que cuando son compiladas y mantenidas por el DBMS puede determinar a todos los estudiantes en libertad condicional. En un DBMS tradicional, un procedimiento explícito el código del programa dural tendría que estar escrito para soportar tales aplicaciones. Pero si

las reglas del mini mundo cambian, generalmente es más conveniente cambiar las reglas declaradas
reglas de deducción que recodificar programas de procedimiento. En la base de datos relacional de hoy
sistemas, es posible asociar **disparadores** con tablas. Un disparador es una forma de regla
activado por las actualizaciones de la tabla, lo que da como resultado la realización de algunas operaciones adicionales
aciones a otras tablas, enviar mensajes, etc. Procedimiento más involucrado
Los procedimientos para hacer cumplir las reglas se denominan popularmente **procedimientos almacenados** ; se vuelven parte de
la definición general de la base de datos y se invocan de manera apropiada cuando ciertas condiciones
ciones se cumplen. Los **sistemas de bases de datos activos** proporcionan una funcionalidad más poderosa ,
que proporcionan reglas activas que pueden iniciar acciones automáticamente cuando ciertos
ocurren eventos y condiciones (consulte el Capítulo 26 para obtener información sobre las bases de datos activas en
Sección 26.1 y bases de datos deductivas en la Sección 26.5).

1.6.10 Implicaciones adicionales del uso el enfoque de base de datos

Esta sección analiza algunas implicaciones adicionales del uso del enfoque de base de datos
que puede beneficiar a la mayoría de las organizaciones.

Potencial para hacer cumplir los estándares. El enfoque de la base de datos permite al DBA
definir y hacer cumplir los estándares entre los usuarios de bases de datos en una organización grande. Esta facilidad
ita la comunicación y cooperación entre varios departamentos, proyectos y
usuarios dentro de la organización. Se pueden definir estándares para nombres y formatos de
elementos de datos, formatos de visualización, estructuras de informes, terminología, etc. El DBA
Puede hacer cumplir los estándares en un entorno de base de datos centralizado más fácilmente que en un
entorno en el que cada grupo de usuarios tiene el control de sus propios archivos de datos y software.

Reducción del tiempo de desarrollo de aplicaciones. Una característica de venta principal de los datos-
enfoque básico es el desarrollo de una nueva aplicación, como la recuperación de ciertos
datos de la base de datos para imprimir un nuevo informe: lleva muy poco tiempo. Diseño
e implementar una gran base de datos multiusuario desde cero puede llevar más tiempo
que escribir una única aplicación de archivo especializada. Sin embargo, una vez que una base de datos está activa
y funcionando, generalmente se requiere mucho menos tiempo para crear nuevas aplicaciones

Página 23

utilizando las instalaciones de DBMS. Se estima que el tiempo de desarrollo usando un DBMS es uno-
sexto a un cuarto de eso para un sistema de archivos.

Flexibilidad. Puede ser necesario cambiar la estructura de una base de datos según sea necesario.
los cambios cambian. Por ejemplo, puede surgir un nuevo grupo de usuarios que necesite información.
actualmente no en la base de datos. En respuesta, puede que sea necesario agregar un archivo al
base de datos o para ampliar los elementos de datos en un archivo existente. Los DBMS modernos permiten
ciertos tipos de cambios evolutivos en la estructura de la base de datos sin afectar-
ing los datos almacenados y los programas de aplicación existentes.

Disponibilidad de información actualizada. Un DBMS hace que la base de datos esté disponible
a todos los usuarios. Tan pronto como se aplique la actualización de un usuario a la base de datos, todos los demás usuarios
puede ver esta actualización inmediatamente. Esta disponibilidad de información actualizada es

esencial para muchas aplicaciones de procesamiento de transacciones, como los sistemas de reserva o bases de datos bancarias, y es posible gracias al control de concurrencia y recuperación subsistemas de un DBMS.

Economías de escala. El enfoque DBMS permite la consolidación de datos y aplicaciones, reduciendo así la cantidad de superposición inútil entre las actividades de personal de procesamiento de datos en diferentes proyectos o departamentos, así como cies entre aplicaciones. Esto permite que toda la organización invierta en más potentes procesadores, dispositivos de almacenamiento o equipos de red, en lugar de tener cada uno departamento compra su propio equipo (de menor rendimiento). Esto reduce en general costos de operación y manejo.

1.7 Breve historia de las aplicaciones de bases de datos

A continuación, ofrecemos una breve descripción histórica de las aplicaciones que utilizan DBMS y cómo estas aplicaciones proporcionaron el ímpetu para nuevos tipos de sistemas de bases de datos.

1.7.1 Aplicaciones de bases de datos tempranas que utilizan jerárquica y sistemas de red

Muchas de las primeras aplicaciones de bases de datos mantenían registros en grandes organizaciones como corporaciones, universidades, hospitales y bancos. En muchas de estas aplicaciones, había un gran número de registros de estructura similar. Por ejemplo, en una universidad aplicación, se mantendrá información similar para cada estudiante, cada curso, cada registro de calificación, y así sucesivamente. También había muchos tipos de registros y muchos interrelaciones entre ellos.

Uno de los principales problemas de los primeros sistemas de bases de datos era la mezcla de relaciones óptimas con el almacenamiento físico y la colocación de registros en disco. Por lo tanto, estos sistemas no proporcionaron suficiente abstracción de datos y datos del programa. capacidades de independencia. Por ejemplo, los registros de calificaciones de un estudiante en particular podría almacenarse físicamente junto al expediente del estudiante. Aunque esto proporcionó

Página 24

acceso eficiente para las consultas y transacciones originales que la base de datos fue diseñado para manejar, no proporcionó suficiente flexibilidad para acceder a los registros de manera eficiente cuando se identificaron nuevas consultas y transacciones. En particular, las nuevas consultas que requirió una organización de almacenamiento diferente para un procesamiento eficiente fueron bastante difíciles para implementar de manera eficiente. También fue laborioso reorganizar la base de datos cuando Se realizaron cambios en los requisitos de la aplicación.

Otra deficiencia de los primeros sistemas era que solo proporcionaban programación interfaces de idioma. Esto hizo que la implementación fuera lenta y costosa nuevas consultas y transacciones, ya que los nuevos programas tenían que ser escritos, probados y depurado. La mayoría de estos sistemas de bases de datos se implementaron en grandes y costosas computadoras mainframe que comenzaron a mediados de la década de 1960 y continuaron

durante las décadas de 1970 y 1980. Los principales tipos de sistemas tempranos se basaron en tres paradigmas principales: sistemas jerárquicos, sistemas basados en modelos de red y sistemas de archivos invertidos.

1.7.2 Proporcionar abstracción de datos y flexibilidad de aplicaciones con bases de datos relacionales

Las bases de datos relacionales se propusieron originalmente para separar el almacenamiento físico de datos de su representación conceptual y para proporcionar una base matemática para la representación y consulta de datos. El modelo de datos relacionales también introdujo lenguajes de consulta de alto nivel que proporcionaron una alternativa al lenguaje de programación interfaces, lo que hace que sea mucho más rápido escribir nuevas consultas. Representación relacional de los datos se parecen un poco al ejemplo que presentamos en la Figura 1.2. Sistema relacional Inicialmente, los sistemas estaban destinados a las mismas aplicaciones que los sistemas anteriores, y gran flexibilidad para desarrollar nuevas consultas rápidamente y reorganizar la base de datos los requisitos cambiaron. Por lo tanto, la abstracción de datos y la independencia de los datos del programa mejoraron mucho en comparación con los sistemas anteriores.

Los primeros sistemas relacionales experimentales se desarrollaron a fines de la década de 1970 y el sistemas de gestión de bases de datos relacionales comerciales (RDBMS) introducidos en el principios de la década de 1980 fueron bastante lentos, ya que no utilizaban punteros de almacenamiento físico o ubicación de registros para acceder a registros de datos relacionados. Con el desarrollo de nuevos técnicas de almacenamiento e indexación y mejor procesamiento y optimización de consultas, su desempeño mejoró. Eventualmente, las bases de datos relacionales se convirtieron en el dominio tipo de sistema de base de datos para aplicaciones de bases de datos tradicionales. Datos relacionales Ahora existen bases en casi todos los tipos de computadoras, desde pequeñas computadoras personales a grandes servidores.

1.7.3 Aplicaciones orientadas a objetos y la necesidad para bases de datos más complejas

La aparición de lenguajes de programación orientados a objetos en la década de 1980 y la La necesidad de almacenar y compartir objetos complejos y estructurados condujo al desarrollo de Bases de datos orientadas a objetos (OODB). Inicialmente, los OODB se consideraban un competidor

Página 25

a las bases de datos relacionales, ya que proporcionaban estructuras de datos más generales. Ellos también incorporó muchos de los paradigmas útiles orientados a objetos, como datos abstractos tipos, encapsulación de operaciones, herencia e identidad de objeto. sin embargo, el La complejidad del modelo y la falta de un estándar temprano contribuyeron a su limitación. uso de ited. Ahora se utilizan principalmente en aplicaciones especializadas, como ingeniería sistemas de diseño, publicación multimedia y fabricación. A pesar de las expectativas que tendrán un gran impacto, su penetración general en el producto de la base de datos El mercado de ucts permanece bajo. Además, se incorporaron muchos conceptos orientados a objetos en las versiones más nuevas de DBMS relacionales, lo que lleva a objetos relacionales sistemas de gestión de bases de datos, conocidos como ORDBMS.

1.7.4 Intercambio de datos en la web para comercio electrónico con XML

La World Wide Web proporciona una gran red de computadoras interconectadas.
Los usuarios pueden crear páginas web estáticas utilizando un lenguaje de publicación web, como Hyper-
Text Markup Language (HTML) y almacene estos documentos en servidores web donde
otros usuarios (clientes) pueden acceder a ellos y verlos a través de navegadores web. Docu-
Los mensajes se pueden vincular a través de **hipervínculos** , que son indicadores de otros documentos.
A partir de la década de 1990, el comercio electrónico (e-commerce) surgió como un importante
aplicación en la Web. Gran parte de la información crítica sobre el comercio electrónico Web
páginas se extraen dinámicamente datos de DBMS, como información de vuelo,
precios de uct y disponibilidad de productos. Se desarrollaron una variedad de técnicas para permitir
el intercambio de datos extraídos dinámicamente en la Web para mostrarlos en la Web
páginas. EXtended Markup Language (XML) es un estándar para intercambiar
datos entre varios tipos de bases de datos y páginas web. XML combina conceptos
a partir de los modelos utilizados en sistemas de documentos con conceptos de modelado de bases de datos.
El capítulo 13 está dedicado a una descripción general de XML.

1.7.5 Ampliación de las capacidades de la base de datos para nuevas aplicaciones

El éxito de los sistemas de bases de datos en aplicaciones tradicionales alentó el desarrollo
operadores de otros tipos de aplicaciones para intentar utilizarlos. Tales aplicaciones
tradicionalmente utilizaba su propio software especializado y estructuras de datos y archivos.
Los sistemas de bases de datos ahora ofrecen extensiones para soportar mejor los requisitos especializados.
mentos para algunas de estas aplicaciones. Los siguientes son algunos ejemplos de estos
aplicaciones:

- Aplicaciones **científicas** que almacenan grandes cantidades de datos resultantes de experimentos en áreas como la física de altas energías, el mapeo del genoma humano y el descubrimiento de estructuras proteicas
- Almacenamiento y recuperación de **imágenes** , incluidas noticias escaneadas o fotografías personales. gráficos, imágenes fotográficas de satélite e imágenes de procedimientos médicos como radiografías y pruebas de resonancia magnética (resonancia magnética)

Página 26

- Almacenamiento y recuperación de **videos** , como películas y **videoclips** de noticias. o cámaras digitales personales
- Aplicaciones de **minería de datos** que analizan grandes cantidades de datos para buscar las ocurrencias de patrones o relaciones específicas, y para identificar patrones inusuales en áreas como la detección de fraudes con tarjetas de crédito
- Aplicaciones **espaciales** que almacenan y analizan ubicaciones espaciales de datos, como información meteorológica, mapas utilizados en los sistemas de información geográfica y sistemas de navegación para automóviles

- Aplicaciones de **series de tiempo** que almacenan información como datos económicos en puntos regulares en el tiempo, como ventas diarias y nacionales brutos mensuales cifras de productos

Rápidamente se hizo evidente que los sistemas relacionales básicos no eran muy adecuados para muchos de estas aplicaciones, generalmente por una o más de las siguientes razones:

- Se necesitaban estructuras de datos más complejas para modelar la aplicación. que la simple representación relacional.
- Se necesitaban nuevos tipos de datos además de los numéricos y de caracteres básicos. tipos de cadenas.
- Fueron necesarias nuevas operaciones y construcciones de lenguaje de consulta para manipular tarde los nuevos tipos de datos.
- Se necesitaban nuevas estructuras de almacenamiento e indexación para una búsqueda eficiente en los nuevos tipos de datos.

Esto llevó a los desarrolladores de DBMS a agregar funcionalidad a sus sistemas. Alguna funcionalidad era de propósito general, como la incorporación de conceptos de datos orientados a objetos bases en sistemas relacionales. Otra funcionalidad tenía un propósito especial, en la forma de módulos opcionales que podrían utilizarse para aplicaciones específicas. Por ejemplo, los usuarios podría comprar un módulo de serie temporal para usarlo con su DBMS relacional para su tiempo aplicación en serie.

1.7.6 Aparición de sistemas de almacenamiento de macrodatos y bases de datos NOSQL

En la primera década del siglo XXI, la proliferación de aplicaciones y plataformas como sitios web de redes sociales, grandes empresas de comercio electrónico, búsqueda web índices, y el almacenamiento / respaldo en la nube llevaron a un aumento en la cantidad de datos almacenados en grandes bases de datos y servidores masivos. Se necesitaban nuevos tipos de sistemas de bases de datos para administrar estas enormes bases de datos, sistemas que proporcionarían búsquedas rápidas y recuperación, así como almacenamiento confiable y seguro de tipos de datos no tradicionales, como publicaciones en redes sociales y tweets. Algunos de los requisitos de estos nuevos sistemas fueron no es compatible con los DBMS relacionales SQL (SQL es el modelo de datos estándar y lenguaje para bases de datos relacionales). El término NOSQL generalmente se interpreta como No Solo SQL, lo que significa que en sistemas que gestionan grandes cantidades de datos, algunos de los los datos se almacenan usando sistemas SQL, mientras que otros datos se almacenarían usando NOSQL, dependiendo de los requisitos de la aplicación.

Página 27

1.8 Cuándo no usar un DBMS

A pesar de las ventajas de usar un DBMS, hay algunas situaciones en las que un El DBMS puede implicar costos generales innecesarios en los que no se incurriría en procesamiento de archivos tradicional. Los costos generales de usar un DBMS se deben a la siguiendo:

- Alta inversión inicial en hardware, software y capacitación.
- La generalidad que proporciona un DBMS para definir y procesar datos
- Gastos generales para proporcionar seguridad, control de simultaneidad, recuperación e integración. funciones rity

Por lo tanto, puede ser más conveniente desarrollar aplicaciones de bases de datos personalizadas. en las siguientes circunstancias:

- Aplicaciones de base de datos simples y bien definidas que no se espera que cambien en absoluto
- Requisitos estrictos en tiempo real para algunos programas de aplicación que pueden no se cumple debido a la sobrecarga de DBMS
- Sistemas integrados con capacidad de almacenamiento limitada, donde un propósito general DBMS no encajaría
- Sin acceso de múltiples usuarios a los datos

Algunas industrias y aplicaciones han optado por no utilizar DBMS. Por ejemplo, muchas herramientas de diseño asistido por computadora (CAD) utilizadas por Los ingenieros civiles y técnicos tienen software de gestión de datos y archivos patentado que está diseñado para las manipulaciones internas de dibujos y objetos 3D. Similar, Los sistemas de comunicación y conmutación diseñados por empresas como AT&T fueron primeras manifestaciones de software de base de datos que se hizo para funcionar muy rápido con datos organizados jerárquicamente para un acceso rápido y enrutamiento de llamadas. Implementación de GIS Las operaciones a menudo implementan sus propios esquemas de organización de datos para implementar funciones relacionadas con el procesamiento de mapas, contornos físicos, líneas, polígonos, etc.

1.9 Resumen

En este capítulo definimos una base de datos como una colección de datos relacionados, donde los datos significa hechos registrados. Una base de datos típica representa algún aspecto del mundo real y se utiliza con fines específicos por uno o más grupos de usuarios. Un DBMS es un paquete de software generalizado para implementar y mantener una computadora base de datos. La base de datos y el software juntos forman un sistema de base de datos. Identificamos Varias características que distinguen el enfoque de la base de datos del tradicional aplicaciones de procesamiento de archivos, y discutimos las principales categorías de bases de datos usuarios, o los actores en la escena. Observamos que además de los usuarios de la base de datos, Hay varias categorías de personal de apoyo, o trabajadores detrás de escena, en un entorno base.

Página 28

Presentamos una lista de capacidades que debe proporcionar el software DBMS para el DBA, los diseñadores de bases de datos y los usuarios finales para ayudarlos a diseñar, administrar y utilizar una base de datos. Luego dimos una breve perspectiva histórica sobre la evolución de los datos. aplicaciones base. Señalamos el rápido crecimiento reciente de las cantidades y tipos de datos que deben almacenarse en bases de datos, y discutimos la aparición de nuevos

sistemas para el manejo de aplicaciones de “big data”. Finalmente, discutimos la sobrecarga costos de usar un DBMS y discutimos algunas situaciones en las que puede no ser Es bueno usar uno.

Preguntas de revisión

1.1. Defina los siguientes términos: datos, base de datos, DBMS, sistema de base de datos, datos-catálogo base, independencia de datos de programa, vista de usuario, DBA, usuario final, enlatado transacción, sistema de base de datos deductivo, objeto persistente, metadatos y aplicación de procesamiento de transacciones.

1.2. ¿Qué cuatro tipos principales de acciones involucran bases de datos? Analice brevemente cada uno.

1.3. Discutir las principales características del enfoque de la base de datos y en qué se diferencia de los sistemas de archivos tradicionales.

1.4. ¿Cuáles son las responsabilidades del DBA y los diseñadores de la base de datos?

1.5. ¿Cuáles son los diferentes tipos de usuarios finales de bases de datos? Discuta las principales actividades lazos de cada uno.

1.6. Analice las capacidades que debería proporcionar un DBMS.

1.7. Discutir las diferencias entre los sistemas de bases de datos y la recuperación de información. sistemas.

Ejercicios

1.8. Identifique algunas consultas informales y actualice las operaciones que esperaría para aplicar a la base de datos que se muestra en la Figura 1.2.

1.9. ¿Cuál es la diferencia entre redundancia controlada y no controlada? Ilustre con ejemplos.

1.10. Especifique todas las relaciones entre los registros de la base de datos que se muestran en Figura 1.2.

1,11. Proporcione algunas vistas adicionales que puedan necesitar otros grupos de usuarios para base de datos que se muestra en la Figura 1.2.

1.12. Cite algunos ejemplos de restricciones de integridad que crea que pueden aplicarse a la base de datos que se muestra en la Figura 1.2.

1,13. Dé ejemplos de sistemas en los que puede tener sentido utilizar archivos tradicionales procesamiento en lugar de un enfoque de base de datos.

Página 29

1,14. Considere la Figura 1.2.

a. Si el nombre del Departamento de 'CS' (Ciencias de la Computación) cambia a 'CSSE' (Informática e Ingeniería de Software) y el departamento
El prefijo de respuesta para el número de curso también cambia, identifique la

umns en la base de datos que deberían actualizarse.

si. ¿Puede reestructurar las columnas en el CURSO , SECCIÓN y
¿ Tablas de PRERREQUISITO para que solo sea necesario actualizar una columna?

Bibliografía seleccionada

El número de octubre de 1991 de Communications of the ACM and Kim (1995) incluye varios artículos que describen DBMS de próxima generación; muchas de las características de la base de datos discutidos en el primero están ahora disponibles comercialmente. El número de marzo de 1976 de ACM Computing Surveys ofrece una introducción temprana a los sistemas de bases de datos y puede proporcionar una perspectiva histórica para el lector interesado. Incluiremos referencias a otros conceptos, sistemas y aplicaciones presentados en este capítulo en la capítulos de texto que tratan cada tema con más detalle.

Esta página se dejó en blanco intencionalmente

capitulo 2

Conceptos del sistema de base de datos y Arquitectura

2.1 Modelos, esquemas e instancias

2.2 Arquitectura de tres esquemas e independencia de datos

2.3 Lenguajes de base de datos y interfaces

2.4 El entorno del sistema de base de datos

2.5 Arquitecturas cliente / servidor centralizadas y de dos niveles

2.6 Arquitecturas de tres niveles y cuatro niveles

2.7 Resumen

La arquitectura de los paquetes DBMS ha evolucionado
El paquete de software DBMS era un sistema estrechamente integrado, para el DBMS moderno
paquetes que son de diseño modular, con una arquitectura de sistema cliente / servidor. los
el crecimiento reciente en la cantidad de datos que requieren almacenamiento ha llevado a sistemas de bases de datos
con arquitecturas distribuidas compuestas por miles de computadoras que administran
los almacenes de datos. Esta evolución refleja las tendencias en informática, donde grandes centros
Las computadoras mainframe tralizadas son reemplazadas por cientos de estaciones de trabajo distribuidas.
ciones y ordenadores personales conectados a través de redes de comunicaciones a varios
tipos de máquinas servidor: servidores web, servidores de bases de datos, servidores de archivos, aplicaciones
servidores, etc. Los entornos actuales de **computación en la nube** constan de miles
arenas de grandes servidores que gestionan los llamados **big data** para los usuarios de la Web.

En una arquitectura DBMS cliente / servidor básica, la funcionalidad del sistema se distribuye
entre dos tipos de módulos. [1] Un **módulo de cliente** suele estar diseñado para que
se ejecutará en un dispositivo móvil, estación de trabajo de usuario o computadora personal (PC). Typi-
En general, los programas de aplicación y las interfaces de usuario que acceden a la base de datos se ejecutan en el
módulo cliente. Por lo tanto, el módulo de cliente maneja la interacción del usuario y proporciona
las interfaces fáciles de usar, como aplicaciones para dispositivos móviles, o formularios o menús
GUI (interfaces gráficas de usuario) basadas en PC. El otro tipo de módulo, llamado
un **módulo de servidor** , generalmente maneja el almacenamiento de datos, el acceso, la búsqueda y otras funciones
ciones. Discutimos las arquitecturas cliente / servidor con más detalle en la Sección 2.5. Primero,
debemos estudiar conceptos más básicos que nos darán una mejor comprensión de
arquitecturas de bases de datos modernas.

[1] Como veremos en la Sección 2.5, existen variaciones en esta simple arquitectura cliente / servidor de dos niveles.

En este capítulo presentamos la terminología y los conceptos básicos que se utilizarán a lo largo del texto. La sección 2.1 analiza los modelos de datos y define los conceptos de esquemas e instancias, que son fundamentales para el estudio del sistema de bases de datos tems. Discutimos la arquitectura DBMS de tres esquemas y la independencia de datos en la Sección 2.2; Esto proporciona una perspectiva del usuario sobre lo que se supone que debe hacer un DBMS. hacer. En la Sección 2.3 describimos los tipos de interfaces y lenguajes que son típicos proporcionado por un DBMS. La sección 2.4 analiza el software del sistema de base de datos ambiente. La sección 2.5 ofrece una descripción general de varios tipos de cliente / servidor arquitecturas. Finalmente, la Sección 2.6 presenta una clasificación de los tipos de DBMS paquetes. La sección 2.7 resume el capítulo.

El material de las Secciones 2.4 a 2.6 proporciona conceptos detallados que pueden considerado como complementario al material básico de introducción.

2.1 Modelos de datos, esquemas e instancias

Una característica fundamental del enfoque de la base de datos es que proporciona nivel de abstracción de datos. **La abstracción de datos** generalmente se refiere a la supresión de detalles de la organización y el almacenamiento de datos, y el resaltado de las características esenciales turas para una mejor comprensión de los datos. Una de las principales características del El enfoque de la base de datos es apoyar la abstracción de datos para que diferentes usuarios puedan percibir datos en su nivel de detalle preferido. Un **modelo de datos:** una colección de conceptos que se puede utilizar para describir la estructura de una base de datos: proporciona los medios necesarios para lograr esta abstracción. [2] Por estructura de una base de datos nos referimos a los tipos de datos, relaciones y restricciones que se aplican a los datos. La mayoría de los modelos de datos también incluyen conjunto de **operaciones básicas** para especificar recuperaciones y actualizaciones en la base de datos.

Además de las operaciones básicas proporcionadas por el modelo de datos, cada vez es más común para incluir conceptos en el modelo de datos para especificar el **aspecto dinámico** o **comportamiento** de una aplicación de base de datos. Esto permite al diseñador de la base de datos especificar un conjunto de operaciones válidas definidas por el usuario que están permitidas en los objetos de la base de datos. [3] An ejemplo de una operación definida por el usuario podría ser COMPUTE_GPA , que puede ser aplicado a un objeto ESTUDIANTE . Por otro lado, operaciones genéricas para insertar, eliminar, modificar o recuperar cualquier tipo de objeto a menudo se incluyen en los datos básicos operaciones modelo. Los conceptos para especificar el comportamiento son fundamentales para la orientación a objetos. modelos de datos (ver el Capítulo 12), pero también se están incorporando en más tradicionales modelos de datos. Por ejemplo, los modelos relacionales de objetos (véase el capítulo 12) amplían los modelo relacional para incluir dichos conceptos, entre otros. En los datos relacionales básicos modelo, hay una provisión para adjuntar comportamiento a las relaciones en forma de persistencia módulos almacenados de tienda, conocidos popularmente como procedimientos almacenados (consulte el Capítulo 10).

[2] A veces, la palabra modelo se usa para denotar una descripción de base de datos o un esquema específico, por ejemplo, el modelo de datos de marketing. No usaremos esta interpretación.

[3] La inclusión de conceptos para describir el comportamiento refleja una tendencia por la cual el diseño de bases de datos y el software Las actividades de diseño se combinan cada vez más en una sola actividad. Tradicionalmente, especificar el comportamiento es asociado con el diseño de software.

Página 33

2.1.1 Categorías de modelos de datos

Se han propuesto muchos modelos de datos, que podemos categorizar según los tipos de conceptos que utilizan para describir la estructura de la base de datos. **De alto nivel** o **Los modelos de datos conceptuales** proporcionan conceptos que se acercan a la forma en que muchos usuarios realizan obtener datos, mientras que **los modelos de datos físicos** o de **bajo nivel** proporcionan conceptos que describen los detalles de cómo se almacenan los datos en los medios de almacenamiento de la computadora, generalmente magnéticos discos. Los conceptos proporcionados por modelos de datos físicos generalmente están pensados para computadoras especialistas, no para usuarios finales. Entre estos dos extremos hay una clase de **representación** (o **implementación**) **modelos de datos** , [4] que proporcionan conceptos que pueden ser fácilmente entendidos por los usuarios finales, pero que no se alejan demasiado de la forma en que se organizan los datos nizado en el almacenamiento de la computadora. Los modelos de datos representativos ocultan muchos detalles de los datos almacenamiento en disco, pero se puede implementar directamente en un sistema informático.

Los modelos de datos conceptuales utilizan conceptos como entidades, atributos y relaciones. Una **entidad** representa un objeto o concepto del mundo real, como un empleado o un proyecto. del minimundo que se describe en la base de datos. Un **atributo** representa algunos propiedad de interés que describe con más detalle una entidad, como el nombre del empleado o salario. Una **relación** entre dos o más entidades representa una asociación entre las entidades, por ejemplo, una relación de trabajo entre un empleado y un proyecto. El capítulo 3 presenta el **modelo entidad-relación,** un popular **modelo de** alto nivel modelo de datos conceptual. El capítulo 4 describe abstracciones adicionales utilizadas para modelado, como generalización, especialización y categorías (tipos de unión).

Los modelos de datos representativos o de implementación son los modelos más utilizados frecuentemente en DBMS comerciales tradicionales. Estos incluyen el **relacional** ampliamente utilizado **modelo de datos** , así como los llamados modelos de datos heredados: la **red** y **modelos jerárquicos,** que se han utilizado ampliamente en el pasado. La parte 3 del texto es dedicado al modelo de datos relacionales y sus limitaciones, operaciones y lenguajes. [5] El estándar SQL para bases de datos relacionales se describe en los Capítulos 6 y 7. Representación Los modelos de datos representacionales representan datos mediante el uso de estructuras de registro y, a veces llamados **modelos de datos basados en registros** .

Podemos considerar el **modelo de datos de objetos** como un ejemplo de una nueva familia de modelos de datos de implementación más cercanos a modelos de datos conceptuales. Un estandar para bases de datos de objetos llamado el modelo de objetos ODMG ha sido propuesto por el Grupo de gestión de datos de objetos (ODMG). Describimos las características generales de bases de datos de objetos y el modelo de objeto propuesto como estándar en el Capítulo 12. Objeto Los modelos de datos también se utilizan con frecuencia como modelos conceptuales de alto nivel, en particular principalmente en el dominio de la ingeniería de software.

Los modelos de datos físicos describen cómo se almacenan los datos como archivos en la computadora enviar información como formatos de registro, orden de registro y rutas de acceso. Un

[4] El término modelo de datos de implementación no es un término estándar; lo hemos introducido para hacer referencia a la disponibilidad modelos de datos capaces en sistemas de bases de datos comerciales.

[5] En los Apéndices D y E se incluye un resumen de los modelos de datos jerárquicos y de red. accesible desde el sitio web del libro.

La ruta de acceso es una estructura de búsqueda que realiza la búsqueda de una base de datos en particular. registros eficientes, como indexación o hash. Analizamos la tecnología de almacenamiento físico niques y estructuras de acceso en los capítulos 16 y 17. Un **índice** es un ejemplo de ruta de acceso que permite el acceso directo a los datos mediante un término de índice o una palabra clave. Es similar al índice al final de este texto, excepto que puede estar organizado en una línea oído, jerárquico (estructurado en árbol), o de alguna otra manera.

Otra clase de modelos de datos se conoce como **modelos de datos autodescriptivos** . Los datos El almacenamiento en sistemas basados en estos modelos combina la descripción de los datos con los propios valores de los datos. En los DBMS tradicionales, la descripción (esquema) está separada clasificado a partir de los datos. Estos modelos incluyen **XML** (consulte el Capítulo 12), así como muchos las **tiendas de valores-clave** y los **sistemas NOSQL** (consulte el Capítulo 24) que se crearon recientemente atado para la gestión de big data.

2.1.2 Esquemas, instancias y estado de la base de datos

En un modelo de datos, es importante distinguir entre la descripción del base de datos y la propia base de datos. La descripción de una base de datos se llama **esquema de base de datos** , que se especifica durante el diseño de la base de datos y no se espera cambiar con frecuencia. [6] La mayoría de los modelos de datos tienen ciertas convenciones para mostrar esquemas como diagramas. [7] Un esquema mostrado se denomina **diagrama de esquema** . Figura 2.1 muestra un diagrama esquemático para la base de datos que se muestra en la Figura 1.2; el diagrama dis- reproduce la estructura de cada tipo de registro, pero no las instancias reales de los registros.

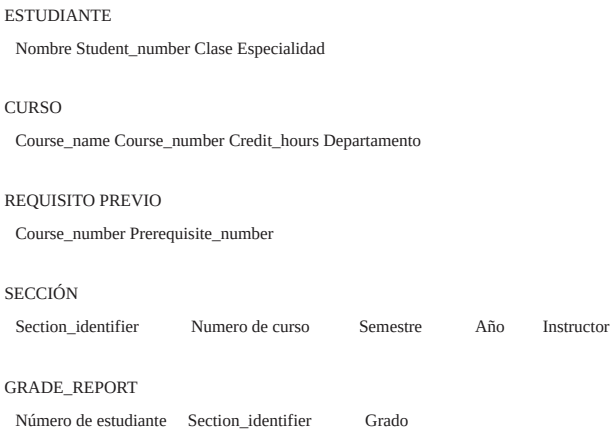


Figura 2.1
Diagrama de esquema para la base de datos en Figura 1.2.

[6] Los cambios de esquema suelen ser necesarios a medida que cambian los requisitos de las aplicaciones de la base de datos. Más Los sistemas de bases de datos incluyen operaciones para permitir cambios de esquema.

[7] En el lenguaje de las bases de datos es habitual usar esquemas como plural para esquema, aunque los esquemas son la forma plural adecuada. La palabra esquema también se usa a veces para referirse a un esquema.

Llamamos a cada objeto del esquema, como ESTUDIANTE o CURSO, un **esquema construir** .

Un diagrama de esquema muestra solo algunos aspectos de un esquema, como los nombres de tipos de registros y elementos de datos, y algunos tipos de restricciones. Otros aspectos no son especificado en el diagrama de esquema; por ejemplo, la Figura 2.1 no muestra los datos tipo de cada elemento de datos ni las relaciones entre los distintos archivos. Muchos tipos de las restricciones no se representan en los diagramas de esquema. Una restricción como los estudiantes con especialización en ciencias de la computación deben tomar CS1310 antes del final de su segundo año El año es bastante difícil de representar en forma de diagrama.

Los datos reales de una base de datos pueden cambiar con bastante frecuencia. Por ejemplo, los datos La base que se muestra en la Figura 1.2 cambia cada vez que agregamos un nuevo estudiante o ingresamos un nuevo grado. Los datos de la base de datos en un momento determinado se denominan **base de datos. estado** o **instantánea** . También se denomina conjunto actual de **ocurrencias** o **instancias** en la base de datos. En un estado de base de datos dado, cada construcción de esquema tiene su propia corriente conjunto de instancias; por ejemplo, la construcción ESTUDIANTE contendrá el conjunto de indicadores entidades de estudiantes individuales (registros) como sus instancias. Muchos estados de bases de datos pueden estructurado para corresponder a un esquema de base de datos particular. Cada vez que insertamos o eliminar un registro o cambiar el valor de un elemento de datos en un registro, cambiamos un estado de la base de datos a otro estado.

La distinción entre el esquema de la base de datos y el estado de la base de datos es muy importante. Cuando **definimos** una nueva base de datos, especificamos su esquema de base de datos solo al DBMS. En este punto, el estado de la base de datos correspondiente es el estado vacío con sin datos. Obtenemos el estado inicial de la base de datos cuando la base de datos es la primera **poblado** o **cargado** con los datos iniciales. A partir de ese momento, cada vez que se actualice La operación se aplica a la base de datos, obtenemos otro estado de la base de datos. En cualquier punto con el tiempo, la base de datos tiene un estado actual. [8] El DBMS es parcialmente responsable de asegurando que cada estado de la base de datos sea un **estado válido** , es decir, un estado que satisface la estructura y las restricciones especificadas en el esquema. Por lo tanto, especifique Es extremadamente importante aplicar un esquema correcto al DBMS y el esquema debe estar diseñado con sumo cuidado. El DBMS almacena las descripciones del esquema. construcciones y restricciones, también llamadas **metadatos,** en el catálogo de DBMS, por lo que que el software DBMS puede consultar el esquema siempre que lo necesite. El esquema a veces se llama **intensión** , y un estado de base de datos se llama una **extensión** de el esquema.

Aunque, como se mencionó anteriormente, no se supone que el esquema cambie con frecuencia, No es infrecuente que los cambios deban aplicarse ocasionalmente al esquema como los requisitos de la aplicación cambian. Por ejemplo, podemos decidir que otro El elemento de datos debe almacenarse para cada registro en un archivo, como agregar la fecha de nacimiento. al esquema del ESTUDIANTE en la Figura 2.1. Esto se conoce como **evolución del esquema** . Más Los DBMS modernos incluyen algunas operaciones para la evolución del esquema que se pueden aplicar mientras la base de datos está operativa.

[8] El estado actual también se denomina instantánea actual de la base de datos. También se le ha llamado base de datos instancia, pero preferimos usar el término instancia para referirnos a registros individuales.

2.2 Arquitectura de tres esquemas e independencia de datos

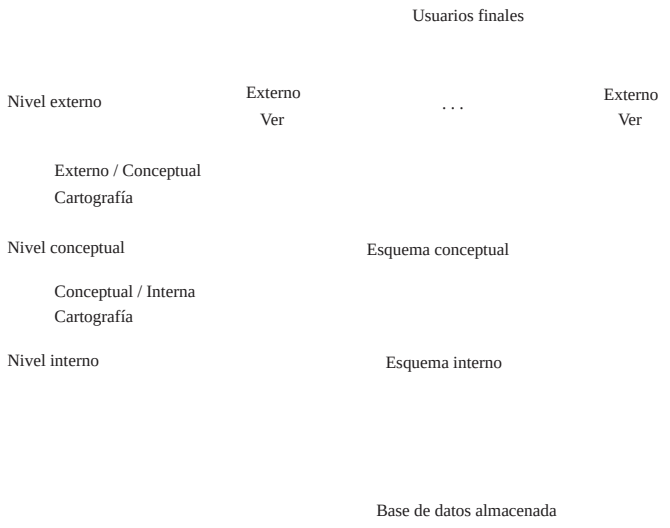
Tres de las cuatro características importantes del enfoque de base de datos, enumeradas en Sección 1.3, son (1) el uso de un catálogo para almacenar la descripción de la base de datos (esquema) de modo que como para que sea autodescriptivo, (2) aislamiento de programas y datos (programa-datos e independencia del funcionamiento del programa) y (3) soporte de múltiples vistas de usuarios. En esta sección especificamos una arquitectura para sistemas de bases de datos, denominada **arquitectura de tres esquemas** , [9] que se propuso para ayudar a lograr y visualizar estas características. Luego, discutimos más el concepto de independencia de datos.

2.2.1 La arquitectura de tres esquemas

El objetivo de la arquitectura de tres esquemas, ilustrada en la Figura 2.2, es separar las aplicaciones de usuario de la base de datos física. En esta arquitectura, los esquemas pueden definirse en los tres niveles siguientes:

1. El **nivel interno** tiene un **esquema interno** , que describe el físico estructura de almacenamiento de la base de datos. El esquema interno utiliza datos físicos modelo y describe los detalles completos de almacenamiento de datos y rutas de acceso para la base de datos.

Figura 2.2
El esquema de tres arquitectura.



[9] Esto también se conoce como ANSI / SPARC (American National Standards Institute / Standards Planning And Requirements Committee), según el comité que la propuso (Tsichritzis & Klug, 1978).

2. El **nivel conceptual** tiene un **esquema conceptual** , que describe la estructura de toda la base de datos para una comunidad de usuarios. El esquema conceptual esconde los detalles de las estructuras de almacenamiento físico y se concentra en describir las entidades vínculos, tipos de datos, relaciones, operaciones de usuario y restricciones. Por lo general, un representante modelo de datos representacionales se utiliza para describir el esquema conceptual cuando un está implementado el sistema de base de datos. Este esquema conceptual de implementación es a menudo se basa en un diseño de esquema conceptual en un modelo de datos de alto nivel.

3. El nivel **externo** o de **vista** incluye varios **esquemas externos** o **usuarios vistas** . Cada esquema externo describe la parte de la base de datos que un participante ular está interesado y oculta el resto de la base de datos de ese grupo de usuario. Como en el nivel anterior, cada esquema externo se implementa típicamente Mentado utilizando un modelo de datos representacional, posiblemente basado en un externo diseño de esquemas en un modelo de datos conceptual de alto nivel.

La arquitectura de tres esquemas es una herramienta conveniente con la que el usuario puede visualizar ize los niveles de esquema en un sistema de base de datos. La mayoría de los DBMS no separan los tres niveles completa y explícitamente, pero apoyan la arquitectura de tres esquemas para cierto punto. Algunos DBMS más antiguos pueden incluir detalles a nivel físico en el concepto. esquema tual. La arquitectura ANSI de tres niveles tiene un lugar importante en la base de datos desarrollo tecnológico porque separa claramente el nivel externo de los usuarios, el el nivel conceptual de la base de datos y el nivel de almacenamiento interno para diseñar una base de datos. Es muy aplicable en el diseño de DBMS, incluso hoy. En la mayoría de los DBMS que apoyar las vistas del usuario, los esquemas externos se especifican en el mismo modelo de datos que describe la información a nivel conceptual (por ejemplo, un DBMS relacional como Oracle o SQLServer utilizan SQL para esto).

Observe que los tres esquemas son solo descripciones de datos; los datos reales se almacenan sólo a nivel físico. En la arquitectura de tres esquemas, cada grupo de usuarios se refiere a su propio esquema externo. Por lo tanto, el DBMS debe transformar una solicitud especificada en un esquema externo en una solicitud contra el esquema conceptual, y luego en una solicitud en el esquema interno para procesar sobre la base de datos almacenada. Si el solicitud es una recuperación de la base de datos, los datos extraídos de la base de datos almacenada deben ser reformateado para que coincida con la vista externa del usuario. Los procesos de transformación las solicitudes y los resultados entre niveles se denominan **asignaciones** . Estas asignaciones pueden ser requiere mucho tiempo, por lo que algunos DBMS, especialmente aquellos que están destinados a admitir pequeños bases de datos: no admiten vistas externas. Incluso en tales sistemas, sin embargo, es necesario Es necesario transformar las solicitudes entre los niveles conceptual e interno.

2.2.2 Independencia de datos

La arquitectura de tres esquemas se puede utilizar para explicar mejor el concepto de **datos independencia** , que se puede definir como la capacidad de cambiar el esquema en un nivel de un sistema de base de datos sin tener que cambiar el esquema en el siguiente nivel superior nivel. Podemos definir dos tipos de independencia de datos:

1. **La independencia lógica de los datos** es la capacidad de cambiar el esquema conceptual sin tener que cambiar esquemas externos o programas de aplicación. Nosotros

puede cambiar el esquema conceptual para expandir la base de datos (agregando un tipo de registro o elemento de datos), para cambiar las restricciones o para reducir la base de datos (eliminando un tipo de registro o elemento de datos). En el último caso, esquemas externos que se refieren solo a los datos restantes no deberían verse afectados. Por ejemplo, El esquema externo de la Figura 1.5 (a) no debe verse afectado por el cambio de GRADE_REPORT (o tipo de registro) que se muestra en la Figura 1.2 en el que se muestra en la Figura 1.6 (a). Solo la definición de la vista y las asignaciones deben cambiarse en un DBMS que admita la independencia lógica de los datos. Después de la El esquema conceptual sufre una reorganización lógica, el pro- Los gramos que hacen referencia a las construcciones del esquema externo deben funcionar como antes. Los cambios en las restricciones se pueden aplicar al esquema conceptual sin afectando los esquemas externos o programas de aplicación.

2. **La independencia de los datos físicos** es la capacidad de cambiar el esquema interno. sin tener que cambiar el esquema conceptual. Por tanto, el esquema externo mas no es necesario cambiar también. Los cambios en el esquema interno pueden ser necesario porque algunos archivos físicos fueron reorganizados, por ejemplo, por creación de estructuras de acceso adicionales para mejorar el rendimiento de la recuperación o actualizar. Si los mismos datos que antes permanecen en la base de datos, no debemos hay que cambiar el esquema conceptual. Por ejemplo, proporcionar un acceso ruta para mejorar la velocidad de recuperación de registros de SECCIÓN (Figura 1.2) por semestres ter y año no deberían requerir una consulta como enumerar todas las secciones ofrecidas en otoño 2008 para ser cambiado, aunque la consulta se ejecutaría de manera más eficiente. por el DBMS utilizando la nueva ruta de acceso.

Generalmente, la independencia de los datos físicos existe en la mayoría de las bases de datos y entornos de archivos. mentos donde detalles físicos, como la ubicación exacta de los datos en el disco, y los detalles de almacenamiento de codificación, ubicación, compresión, división, fusión de registros, etc., están ocultos para el usuario. Las aplicaciones desconocen estos detalles. Por otro lado, la independencia lógica de los datos es más difícil de lograr porque permite cambios estructurales y de restricciones sin afectar los programas de aplicación, una requisito mucho más estricto.

Siempre que tengamos un DBMS de varios niveles, su catálogo debe ampliarse para incluir información sobre cómo mapear solicitudes y datos entre los distintos niveles. El DBMS utiliza software adicional para realizar estas asignaciones haciendo referencia a la asignación información en el catálogo. La independencia de los datos se produce porque cuando el esquema es cambiado en algún nivel, el esquema en el siguiente nivel superior permanece sin cambios; solamente se cambia el mapeo entre los dos niveles. Por tanto, los programas de aplicación No es necesario cambiar el anillo al esquema de nivel superior.

2.3 Idiomas e interfaces de la base de datos

En la Sección 1.4 discutimos la variedad de usuarios soportados por un DBMS. El DBMS debe proporcionar lenguajes e interfaces apropiados para cada categoría de usuarios. En En esta sección discutimos los tipos de lenguajes e interfaces proporcionados por un DBMS. y las categorías de usuarios a las que se dirige cada interfaz.

2.3.1 Idiomas DBMS

Una vez que se completa el diseño de una base de datos y se elige un DBMS para implementar el base de datos, el primer paso es especificar esquemas conceptuales e internos para los datos- base y cualquier mapeo entre los dos. En muchos DBMS donde no hay separaciones estrictas Se mantiene la ción de niveles, un lenguaje, llamado **lenguaje de definición de datos.** (**DDL**), es utilizado por el DBA y por los diseñadores de bases de datos para definir ambos esquemas. los DBMS tendrá un compilador DDL cuya función es procesar declaraciones DDL en para identificar descripciones de las construcciones del esquema y almacenar el esquema descripción en el catálogo DBMS.

En los DBMS donde se mantiene una clara separación entre lo conceptual y lo niveles internos, el DDL se utiliza para especificar solo el esquema conceptual. Otro lenguaje, el **lenguaje de definición de almacenamiento** (**SDL**), se utiliza para especificar el esquema. Las asignaciones entre los dos esquemas se pueden especificar en cualquiera de estos idiomas. En la mayoría de los DBMS relacionales de hoy, no existe un lenguaje específico que realiza el papel de SDL. En cambio, el esquema interno se especifica mediante una combinación de funciones, parámetros y especificaciones relacionadas con el almacenamiento de archivos. Estos permiten el personal de DBA para controlar las opciones de indexación y la asignación de datos al almacenamiento. Por un verdadero arquitectura de tres esquemas, necesitaríamos un tercer lenguaje, la **definición de vista lenguaje** (**VDL**), para especificar las vistas del usuario y sus asignaciones al concepto esquema, pero en la mayoría de los DBMS, el DDL se utiliza para definir tanto conceptual como externo esquemas. En los DBMS relacionales, SQL se utiliza en el rol de VDL para definir usuarios o **vistas de la** aplicación como resultados de consultas predefinidas (véanse los Capítulos 6 y 7).

Una vez que se compilan los esquemas de la base de datos y la base de datos se llena con datos, los usuarios deben tener algunos medios para manipular la base de datos. Manipulaciones típicas incluyen la recuperación, inserción, eliminación y modificación de los datos. El programa DBMS vides un conjunto de operaciones o un lenguaje llamado **lenguaje de manipulación de datos** (**DML**) para estos fines.

En los DBMS actuales, los tipos de lenguajes anteriores no se suelen considerar distribuidos lenguas tintas; en su lugar, se utiliza un lenguaje integral integral que incluye construcciones para la definición del esquema conceptual, la definición de la vista y la manipulación de datos ción. La definición de almacenamiento generalmente se mantiene separada, ya que se usa para definir estructuras de almacenamiento cal para ajustar el rendimiento del sistema de base de datos, que es generalmente realizado por el personal de DBA. Un ejemplo típico de un lenguaje de base de datos completo guage es el lenguaje de base de datos relacional SQL (vea los Capítulos 6 y 7), que representa envía una combinación de DDL, VDL y DML, así como declaraciones de restricción especificación, evolución del esquema y muchas otras características. El SDL era un componente nent en las primeras versiones de SQL, pero se ha eliminado del lenguaje para mantenerlo en los niveles conceptual y externo solamente.

Hay dos tipos principales de DML. Un LMD de **alto nivel** o sin **procedimiento** puede ser se utiliza por sí solo para especificar operaciones complejas de bases de datos de forma concisa. Muchos DBMS Permitir que las declaraciones DML de alto nivel se ingresen de forma interactiva desde una pantalla monitor o terminal o para integrarse en un lenguaje de programación de propósito general calibre. En el último caso, las declaraciones DML deben identificarse dentro del programa para

Página 40

que pueden ser extraídos por un precompilador y procesados por el DBMS. Un **bajo** DML de **nivel** o de **procedimiento** debe estar integrado en una programación de propósito general idioma. Este tipo de DML generalmente recupera registros u objetos individuales de la base de datos y procesa cada uno por separado. Por lo tanto, necesita usar programación construcciones de lenguaje, como bucles, para recuperar y procesar cada registro de un conjunto de registros. Los DML de bajo nivel también se denominan DML de **registro a la vez** debido a esto propiedad. Los DML de alto nivel, como SQL, pueden especificar y recuperar muchos registros en una sola declaración DML; por lo tanto, se les llama **set-at-a-time** o **set-orientado** DML. Una consulta en un DML de alto nivel a menudo especifica qué datos recuperar en lugar de que cómo recuperarlo; por lo tanto, estos lenguajes también se denominan **declarativos** .

Siempre que los comandos DML, ya sean de alto o bajo nivel, están integrados en un Lenguaje de programación de propósito general, ese lenguaje se llama **lenguaje anfitrión.** y el DML se denomina **sublenguaje de datos** . [10] Por otro lado, un alto nivel El DML utilizado de forma interactiva independiente se denomina **lenguaje de consulta** . En gen- En general, los comandos de recuperación y actualización de un DML de alto nivel se pueden utilizar entre de forma activa y, por tanto, se consideran parte del lenguaje de consulta. [11]

Los usuarios finales ocasionales suelen utilizar un lenguaje de consulta de alto nivel para especificar sus solicitudes, mientras que los programadores usan el DML en su forma incrustada. Para ingenuo y paramet usuarios ricos, por lo general hay **interfaces fáciles de usar** para interactuar con los datos base; Estos también pueden ser utilizados por usuarios ocasionales u otros que no quieran aprender el detalles de un lenguaje de consulta de alto nivel. A continuación, discutimos estos tipos de interfaces.

2.3.2 Interfaces DBMS

Las interfaces fáciles de usar proporcionadas por un DBMS pueden incluir lo siguiente:

Interfaces basadas en menús para clientes web o navegación. Estas interfaces presen- Entra al usuario con listas de opciones (llamadas **menús)** que guían al usuario a través de la mulación de una solicitud. Los menús eliminan la necesidad de memorizar los comandos y sintaxis de un lenguaje de consulta; más bien, la consulta se compone paso a paso paso seleccionando opciones de un menú que se muestra en el sistema. Derribar Los menús son una técnica muy popular en **las interfaces de usuario basadas** en **web** . Ellos son también Se utiliza a menudo en **interfaces de navegación** , que permiten al usuario mirar a través de los contenidos. de una base de datos de forma exploratoria y no estructurada.

Aplicaciones para dispositivos móviles. Estas interfaces ofrecen a los usuarios móviles acceso a sus datos. Por ejemplo, compañías bancarias, de reservas y de seguros, entre muchos otros, proporcionan aplicaciones que permiten a los usuarios acceder a sus datos a través de un dispositivo móvil teléfono o dispositivo móvil. Las aplicaciones tienen interfaces programadas integradas que normalmente

[10] En las bases de datos de objetos, los subidiomas de host y datos suelen formar un lenguaje integrado, para ejemplo, C ++ con algunas extensiones para admitir la funcionalidad de la base de datos. Algunos sistemas relacionales también proporcionar lenguajes integrados, por ejemplo, PL / SQL de Oracle.

[11] De acuerdo con el significado en inglés de la palabra query, realmente debería usarse para describir recuperaciones solo, no actualizaciones.

Página 41

permitir que los usuarios inicien sesión con su nombre de cuenta y contraseña; las aplicaciones luego proporcionan un menú limitado de opciones para el acceso móvil a los datos del usuario, así como opciones como como pagar facturas (para bancos) o hacer reservas (para sitios web de reservas).

Interfaces basadas en formularios. Una interfaz basada en formularios muestra un formulario a cada usuario. Los usuarios pueden completar todas las entradas del **formulario** para insertar nuevos datos, o pueden completar solo ciertas entradas, en cuyo caso el DBMS recuperará datos coincidentes para el resto ing entradas. Los formularios generalmente se diseñan y programan para usuarios ingenuos como caras a transacciones enlatadas. Muchos DBMS tienen **lenguajes de especificación de formularios** , que son lenguajes especiales que ayudan a los programadores a especificar tales formas. Formularios SQL * es un lenguaje basado en formularios que especifica consultas utilizando un formulario diseñado en conjunto ción con el esquema de base de datos relacional. Oracle Forms es un componente de Ora- suite de productos cle que proporciona un amplio conjunto de características para diseñar y construir aplicaciones que utilizan formularios. Algunos sistemas tienen utilidades que definen una forma al permitir el usuario final construye de forma interactiva un formulario de muestra en la pantalla.

Interfaces gráficas de usuario. Una GUI normalmente muestra un esquema al usuario en dia- forma gramatical. Luego, el usuario puede especificar una consulta manipulando el diagrama. En muchos casos, las GUI utilizan tanto menús como formularios.

Interfaces de lenguaje natural. Estas interfaces aceptan solicitudes escritas en inglés. lish o algún otro idioma e intente comprenderlos. Un lenguaje natural La interfaz generalmente tiene su propio esquema, que es similar a la base de datos conceptual esquema, así como un diccionario de palabras importantes. La interfaz de lenguaje natural se refiere a las palabras en su esquema, así como al conjunto de palabras estándar en su dic- ción, que se utilizan para interpretar la solicitud. Si la interpretación tiene éxito, el La interfaz genera una consulta de alto nivel correspondiente a la solicitud de lenguaje natural. y lo envía al DBMS para su procesamiento; de lo contrario, se inicia un diálogo con el usuario para aclarar la solicitud.

Búsqueda de base de datos basada en palabras clave. Son algo similares a la búsqueda web motores, que aceptan cadenas de palabras en lenguaje natural (como inglés o español) y compararlos con documentos en sitios específicos (para motores de búsqueda locales) o en la Web. páginas en la Web en general (para motores como Google o Ask). Usan predefinidos índices de palabras y utilizan funciones de clasificación para recuperar y presentar documentos resultantes menciones en un grado decreciente de coincidencia. Estas interfaces de consulta textual de "forma libre" son aún no es común en las bases de datos relacionales estructuradas, aunque un área de investigación llamada **La consulta basada en palabras clave** ha surgido recientemente para bases de datos relacionales.

Entrada y salida de voz. Uso limitado de voz como consulta de entrada y voz como respuesta a una pregunta o resultado de una solicitud se está convirtiendo en algo común. Aplicación caciones con vocabularios limitados, como consultas de directorio telefónico, llegada / salida, y la información de la cuenta de la tarjeta de crédito, permiten el habla para entrada y salida para permitir a los clientes acceder a esta información. La entrada de voz se detecta utilizando una biblioteca de palabras predefinidas y se utiliza para configurar los parámetros que se suministran a las consultas. Para la salida, una conversión similar de texto o número se llevan a cabo cambios en el habla.

Página 42

Interfaces para usuarios paramétricos. Los usuarios paramétricos, como los cajeros de banco, a menudo tienen un pequeño conjunto de operaciones que deben realizar repetidamente. Por ejemplo, un El cajero puede utilizar teclas de función única para invocar transacciones rutinarias y repetitivas. como depósitos o retiros de cuentas, o consultas de saldo. Analistas de sistemas y Los programadores diseñan e implementan una interfaz especial para cada clase conocida de usuarios ingenuos. Por lo general, se incluye un pequeño conjunto de comandos abreviados, con el objetivo de minimizar el número de pulsaciones de teclas necesarias para cada solicitud.

Interfaces para el DBA. La mayoría de los sistemas de bases de datos contienen comandos privilegiados que solo puede utilizar el personal de DBA. Estos incluyen comandos para crear cuentas, configuración de parámetros del sistema, concesión de autorización de cuenta, cambio de esquema y reorganización de las estructuras de almacenamiento de una base de datos.

2.4 El entorno del sistema de base de datos

Un DBMS es un sistema de software complejo. En esta sección discutimos los tipos de software componentes de software que constituyen un DBMS y los tipos de software del sistema informático productos con los que interactúa el DBMS.

2.4.1 Módulos de componentes DBMS

La figura 2.3 ilustra, de forma simplificada, los componentes típicos del DBMS. los La figura se divide en dos partes. La parte superior de la figura se refiere a los distintos usuarios del entorno de la base de datos y sus interfaces. La parte inferior muestra la módulos internos del DBMS responsables del almacenamiento de datos y procesamiento de actas.

La base de datos y el catálogo DBMS generalmente se almacenan en disco. Acceso al disco está controlado principalmente por el **sistema operativo** (**SO**), que programa el disco leer escribir. Muchos DBMS tienen su propio módulo de **gestión de búfer** para programar Lectura / escritura de disco ule, porque la gestión del almacenamiento en búfer tiene un considerable efecto sobre el rendimiento. La reducción de la lectura / escritura del disco mejora considerablemente el rendimiento. erably. Un módulo de **administrador de datos almacenados de** nivel superior del DBMS controla el acceso a la información DBMS que se almacena en el disco, ya sea que sea parte de la base de datos o el catalogo.

Consideremos primero la parte superior de la Figura 2.3. Muestra interfaces para el personal de DBA, usuarios ocasionales que trabajan con interfaces interactivas para formular consultas, aplicaciones programadores que crean programas utilizando algunos lenguajes de programación host, y Los usuarios paramétricos que realizan la entrada de datos funcionan proporcionando parámetros a predefinidos. actas. El personal de DBA trabaja para definir la base de datos y ajustarla cambios en su definición usando el DDL y otros comandos privilegiados.

El compilador DDL procesa las definiciones de esquema, especificadas en el DDL, y almacena descripciones de los esquemas (metadatos) en el catálogo DBMS. El catalogo incluye información como los nombres y tamaños de archivos, nombres y tipos de datos de elementos de datos, detalles de almacenamiento de cada archivo, información de mapeo entre esquemas y restricciones.

Página 43

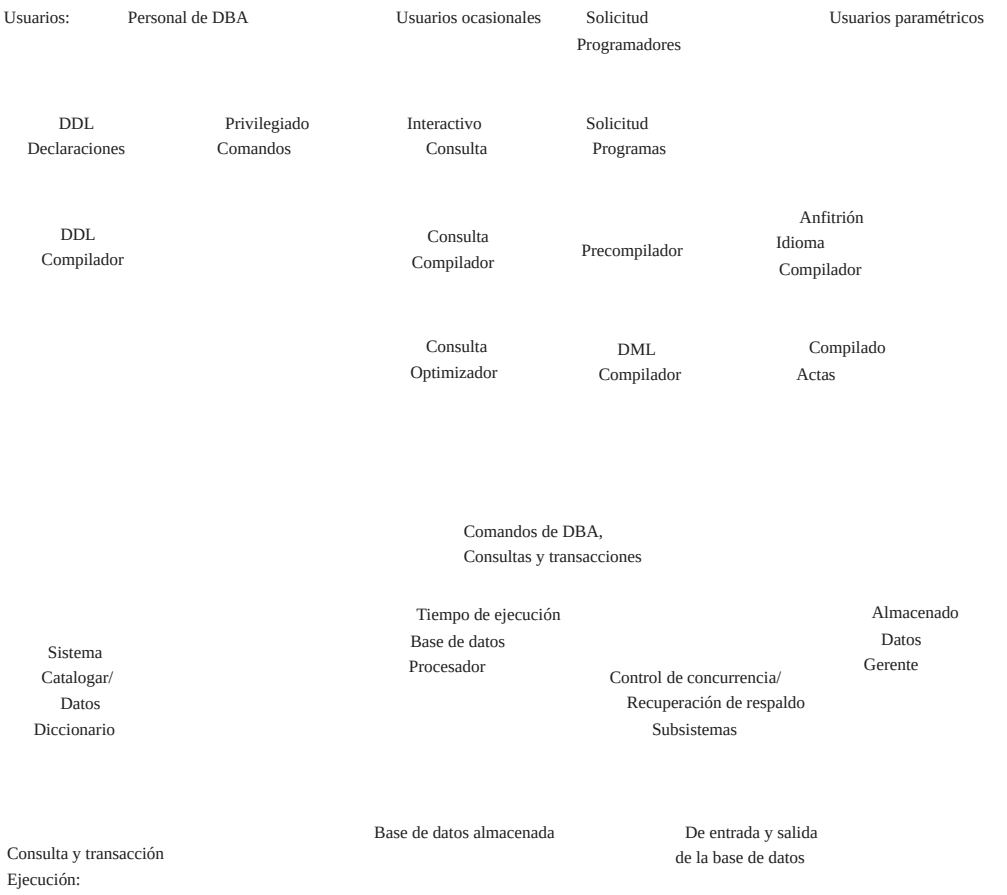


Figura 2.3
Módulos componentes de un DBMS y sus interacciones.

Además, el catálogo almacena muchos otros tipos de información que necesitan los módulos DBMS, que luego pueden buscar la información del catálogo según sea necesario.

Usuarios ocasionales y personas con necesidad ocasional de información de la base de datos interactuar usando la interfaz de **consulta** interactiva en la Figura 2.3. No tenemos explícitamente se muestran las interacciones móviles o basadas en menús o formularios que se utilizan normalmente para generar la consulta interactiva automáticamente o para acceder a transacciones enlatadas. Estas consultas se analizan y validan para verificar que la sintaxis de la consulta sea correcta, nombres de archivos y elementos de datos, y así sucesivamente mediante un **compilador de consultas** que compila

Página 44

ellos en una forma interna. Esta consulta interna está sujeta a optimización de consultas. (discutido en los Capítulos 18 y 19). Entre otras cosas, el **optimizador de consultas** es preocupados por la reordenación y posible reordenamiento de las operaciones, ción de redundancias y uso de algoritmos de búsqueda eficientes durante la ejecución. Eso consulta el catálogo del sistema para obtener información estadística y física sobre el almacena datos y genera código ejecutable que realiza las operaciones necesarias para la consulta y realiza llamadas en el procesador en tiempo de ejecución.

Los programadores de aplicaciones escriben programas en lenguajes host como Java, C o C ++ que se envían a un precompilador. El **precompilador** extrae comandos DML desde un programa de aplicación escrito en un lenguaje de programación anfitrión. Estos com- Las órdenes se envían al compilador DML para su compilación en código objeto para la base de datos acceso. El resto del programa se envía al compilador del lenguaje anfitrión. El objeto Los códigos para los comandos DML y el resto del programa están vinculados, formando un transacción enlatada cuyo código ejecutable incluye llamadas a la base de datos en tiempo de ejecución procesador. También es cada vez más común utilizar lenguajes de programación como como PHP y Python para escribir programas de base de datos. Las transacciones enlatadas se ejecutan repetidamente por usuarios paramétricos a través de PC o aplicaciones móviles; estos usuarios simplemente proporcionan los parámetros de las transacciones. Cada ejecución se considera una transacción. Un ejemplo es una transacción de pago bancaria donde el número de cuenta, beneficiario, y la cantidad se pueden proporcionar como parámetros.

En la parte inferior de la Figura 2.3, el **procesador de base de datos en tiempo de ejecución** ejecuta (1) el comandos privilegiados, (2) los planes de consulta ejecutables y (3) las transacciones enlatadas ciones con parámetros de tiempo de ejecución. Funciona con el **catálogo** del **sistema** y puede actualizarlo. con estadísticas. También funciona con el **administrador de datos almacenados** , que a su vez utiliza servicios del sistema operativo para realizar entradas / salidas de bajo nivel (lectura / escritura) operaciones entre el disco y la memoria principal. El procesador de base de datos en tiempo de ejecución maneja otros aspectos de la transferencia de datos, como la gestión de búferes en la memoria. Algunos DBMS tienen su propio módulo de gestión de búfer, mientras que otros dependen del sistema operativo para la gestión del búfer. Hemos mostrado **control de concurrencia** y **los sistemas de respaldo y recuperación por** separado como un módulo en esta figura. Son integrado en el funcionamiento del procesador de base de datos en tiempo de ejecución para fines de Gestión de transacciones.

Es común que el **programa cliente** que accede al DBMS se ejecute en un equipo o dispositivo separado del equipo en el que reside la base de datos. los el primero se denomina **computadora cliente que** ejecuta el software cliente DBMS y el segundo es llamado **servidor de base de datos** . En muchos casos, el cliente accede a una computadora intermedia, llamado **servidor de aplicaciones** , que a su vez accede al servidor de la base de datos. Elaboramos calificar sobre este tema en la Sección 2.5.

La figura 2.3 no pretende describir un DBMS específico; más bien, ilustra los típicos Módulos DBMS. El DBMS interactúa con el sistema operativo cuando accede al disco: a la base de datos o al catálogo — son necesarios. Si el sistema informático es compartido por muchos usuarios, el sistema operativo programará las solicitudes de acceso al disco DBMS y el procesamiento de DBMS junto con otros procesos. Por otro lado, si el sistema informático es principalmente dedicado a ejecutar el servidor de la base de datos, el DBMS controlará la memoria principal

Página 45

almacenamiento en búfer de páginas de disco. El DBMS también interactúa con compiladores para general-
lenguajes de programación de host de propósito, y con servidores de aplicaciones y pro-
gramos que se ejecutan en máquinas independientes a través de la interfaz de red del sistema.

2.4.2 Utilidades del sistema de base de datos

Además de poseer los módulos de software que se acaban de describir, la mayoría de los DBMS tienen **utilidades de base de datos** que ayudan al DBA a administrar el sistema de base de datos. Utilidad común los lazos tienen los siguientes tipos de funciones:

- **Cargando.** Se utiliza una utilidad de carga para cargar archivos de datos existentes, como texto archivos o archivos secuenciales, en la base de datos. Por lo general, la (fuente) actual para-
  mat del archivo de datos y la estructura de archivo de base de datos deseada (destino) se especifican a la utilidad, que luego reformatea automáticamente los datos y los almacena en la base de datos. Con la proliferación de DBMS, la transferencia de datos desde un DBMS a otro se está volviendo común en muchas organizaciones. Algunos Los proveedores ofrecen **herramientas de conversión** que generan el pro-
  gramos, dadas las descripciones de almacenamiento de la base de datos de origen y destino existente (esquemas internos).
- **Copia de seguridad.** Una utilidad de respaldo crea una copia de respaldo de la base de datos, generalmente por volcar toda la base de datos en cinta u otro medio de almacenamiento masivo. los La copia de seguridad se puede utilizar para restaurar la base de datos en caso de un disco catastrófico. fracaso. Las copias de seguridad incrementales también se utilizan a menudo, donde solo cambian desde se registra la copia de seguridad anterior. La copia de seguridad incremental es más compleja, pero ahorra espacio de almacenamiento.
- **Reorganización del almacenamiento de la base de datos.** Esta utilidad se puede utilizar para reorganizar un conjunto de archivos de base de datos en diferentes organizaciones de archivos y crear un nuevo acceso caminos para mejorar el rendimiento.
- **Supervisión del desempeño.** Tal utilidad monitorea el uso de la base de datos y envía estadísticas al DBA. El DBA utiliza las estadísticas para tomar decisiones. como si reorganizar o no archivos o si agregar o quitar índices para mejorar el rendimiento.

Otras utilidades pueden estar disponibles para ordenar archivos, manejar la compresión de datos, monitorear el acceso de los usuarios, interactuar con la red y realizar otras funciones.

2.4.3 Herramientas, entornos de aplicación, e instalaciones de comunicaciones

Otras herramientas suelen estar disponibles para los diseñadores de bases de datos, los usuarios y el DBMS. CASO Las herramientas [12] se utilizan en la fase de diseño de los sistemas de bases de datos. Otra herramienta que puede ser bastante útil en organizaciones grandes es un **diccionario de datos** ampliado (o **repositorio de datos**)

[12] Aunque CASE significa ingeniería de software asistida por computadora, muchas herramientas CASE se utilizan principalmente para el diseño de bases de datos.

Página 46

sistema . Además de almacenar información de catálogo sobre esquemas y restricciones, el diccionario de datos almacena otra información, como decisiones de diseño, normas de uso dards, descripciones de programas de aplicación e información de usuario. Tal sistema es también llamado **depósito de información** . Se puede acceder a esta información directamente por usuarios o el DBA cuando sea necesario. Una utilidad de diccionario de datos es similar al DBMS catálogo, pero incluye una variedad más amplia de información y se accede principalmente por usuarios en lugar del software DBMS.

Entornos de desarrollo de aplicaciones , como PowerBuilder (Sybase) o JBuilder (Borland), han sido bastante populares. Estos sistemas proporcionan un entorno ment para desarrollar aplicaciones de bases de datos e incluyen instalaciones que ayudan en muchos facetas de los sistemas de bases de datos, incluido el diseño de bases de datos, el desarrollo de GUI, las consultas y actualización y desarrollo de programas de aplicación.

El DBMS también necesita interactuar con el **software de comunicaciones** , cuya función es permitir que los usuarios en ubicaciones remotas del sitio del sistema de base de datos accedan al base de datos a través de terminales de computadora, estaciones de trabajo o computadoras personales. Estas están conectados al sitio de la base de datos a través de hardware de comunicaciones de datos como Enrutadores de Internet, líneas telefónicas, redes de larga distancia, redes locales o comunicaciones por satélite. dispositivos de comunicación. Muchos sistemas de bases de datos comerciales tienen comunicación paquetes que funcionan con el DBMS. El DBMS integrado y la comunicación de datos El sistema de operaciones se llama sistema **DB / DC** . Además, algunos DBMS distribuidos son distribuidos físicamente en varias máquinas. En este caso, la red de comunicaciones Se necesitan obras para conectar las máquinas. Suelen ser **redes de área local. (LAN)** , pero también pueden ser otros tipos de redes.

2.5 Centralizado y Cliente / Servidor Arquitecturas para DBMS

2.5.1 Arquitectura de DBMS centralizada

Las arquitecturas para DBMS han seguido tendencias similares a las de las arquitecturas de sistemas informáticos. Las arquitecturas más antiguas usaban computadoras mainframe para pro- vide el procesamiento principal para todas las funciones del sistema, incluida la aplicación del usuario programas y programas de interfaz de usuario, así como toda la funcionalidad DBMS. los La razón era que en los sistemas más antiguos, la mayoría de los usuarios accedían al DBMS a través de una terminal de computadora. minals que no tenían capacidad de procesamiento y solo proporcionaban capacidades de visualización. Por lo tanto, todo el procesamiento se realizó de forma remota en la carcasa del sistema informático el DBMS, y solo la información de visualización y los controles se enviaron desde la computadora a los terminales de pantalla, que estaban conectados al ordenador central a través de varios tipos de redes de comunicaciones.

A medida que los precios del hardware disminuyeron, la mayoría de los usuarios reemplazaron sus terminales con PC y estaciones de trabajo y, más recientemente, con dispositivos móviles. Al principio, los sistemas de bases de datos utilizaron estas computadoras de manera similar a como habían usado terminales de pantalla, de modo que el El propio DBMS seguía siendo un DBMS **centralizado** en el que toda la funcionalidad del DBMS,

Página 47

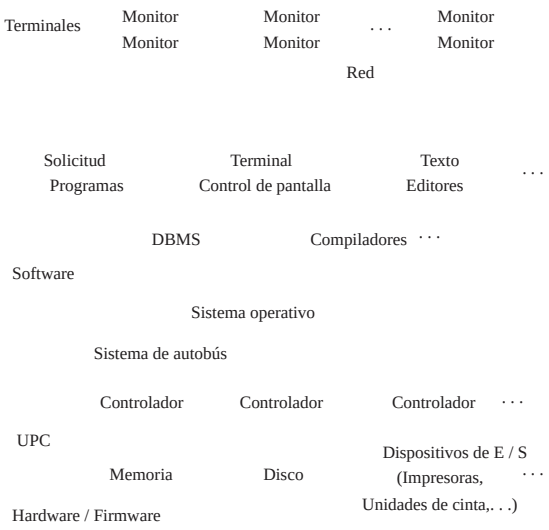


Figura 2.4
Un fisico centralizado
arquitectura.

La ejecución del programa de aplicación y el procesamiento de la interfaz de usuario se llevaron a cabo en una máquina. La figura 2.4 ilustra los componentes físicos en un archivo centralizado. tecture. Gradualmente, los sistemas DBMS comenzaron a explotar la potencia de procesamiento disponible en el lado del usuario, lo que condujo a arquitecturas DBMS cliente / servidor.

2.5.2 Arquitecturas básicas de cliente / servidor

Primero, discutimos la arquitectura cliente / servidor en general; luego discutimos como es aplicado a los DBMS. La **arquitectura cliente / servidor** se desarrolló para hacer frente a las poner entornos en los que una gran cantidad de PC, estaciones de trabajo, servidores de archivos, impresoras, servidores de bases de datos, servidores web, servidores de correo electrónico y otro software y los equipos están conectados a través de una red. La idea es definir **servidores especializados** con funcionalidades específicas. Por ejemplo, es posible conectar varias PC o pequeñas estaciones de trabajo como clientes a un **servidor de archivos** que mantiene los archivos del cliente máquinas. Otra máquina puede ser designado como un **servidor de impresión** al ser con- conectado a varias impresoras; todas las solicitudes de impresión de los clientes se envían a este máquina. **Los servidores web** o los **servidores de correo electrónico** también se incluyen en la categoría de servidores especializados. sangriento. Muchos clientes pueden acceder a los recursos proporcionados por servidores especializados. máquinas. Las **máquinas cliente** proporcionan al usuario las interfaces adecuadas para utilizar estos servidores, así como con potencia de procesamiento local para ejecutar aplicaciones locales. Este concepto puede trasladarse a otros paquetes de software, con pro- gramos, como un paquete CAD (diseño asistido por computadora), que se almacena en

máquinas servidor y ser accesible para múltiples clientes. La figura 2.5 ilustra

Página 48

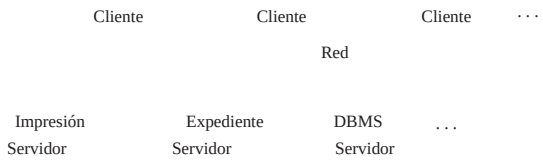


Figura 2.5
Lógico de dos niveles
Servidor de cliente
arquitectura.

arquitectura cliente / servidor a nivel lógico; La figura 2.6 es un diagrama simplificado que muestra la arquitectura física. Algunas máquinas serían solo sitios de clientes (para ejemplo, dispositivos móviles o estaciones de trabajo / PC que solo tienen software de cliente instalado). Otras máquinas serían servidores dedicados y otras tendrían ambos funcionalidad de cliente y servidor.

El concepto de arquitectura cliente / servidor asume un marco subyacente que consta de muchas PC / estaciones de trabajo y dispositivos móviles, así como un número menor de máquinas servidor, conectadas a través de redes inalámbricas o LAN y otros tipos de Red de computadoras. Un **cliente** en este marco es típicamente una máquina de usuario que proporciona vides capacidades de interfaz de usuario y procesamiento local. Cuando un cliente requiere acceso a funcionalidad adicional, como el acceso a la base de datos, que no existe en el ent, se conecta a un servidor que proporciona la funcionalidad necesaria. Un **servidor** es un sistema Elemento que contiene tanto hardware como software que puede proporcionar servicios al cliente. máquinas, como acceso a archivos, impresión, archivo o acceso a bases de datos. En general, algunas máquinas instalan solo software de cliente, otras solo software de servidor, y aún así otros pueden incluir software tanto de cliente como de servidor, como se ilustra en la Figura 2.6. Sin embargo, es más común que el software del cliente y del servidor se ejecuten normalmente en

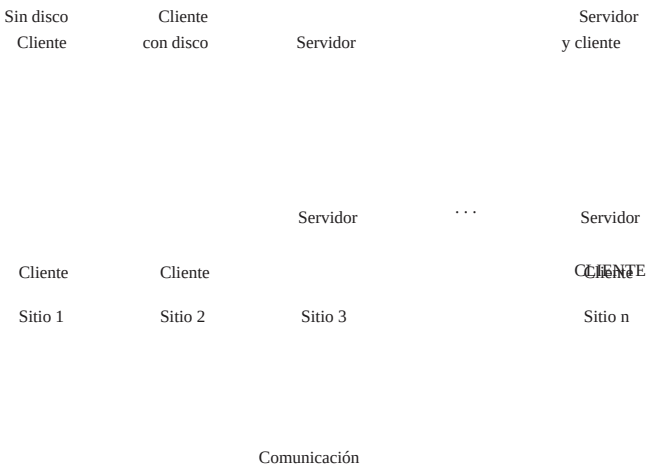


Figura 2.6
Físico de dos niveles
Servidor de cliente
arquitectura.

Página 49

máquinas. Se crearon dos tipos principales de arquitecturas DBMS básicas en este marco cliente / servidor mentiroso: **dos niveles** y **tres niveles** . [13] Los discutimos a continuación.

2.5.3 Arquitecturas cliente / servidor de dos niveles para DBMS

En los sistemas de administración de bases de datos relacionales (RDBMS), muchos de los cuales comenzaron como sistemas centralizados, los componentes del sistema que se movieron primero al del lado del cliente eran la interfaz de usuario y los programas de aplicación. Porque SQL (ver Los capítulos 6 y 7) proporcionaron un lenguaje estándar para RDBMS, esto creó un punto lógico de división entre cliente y servidor. Por tanto, la consulta y transacción La funcionalidad de instalación relacionada con el procesamiento de SQL permaneció en el lado del servidor. En tal arquitectura, el servidor a menudo se denomina **servidor de consultas** o **transacción servidor** porque proporciona estas dos funcionalidades. En un RDBMS, el servidor es también llamado **servidor SQL** .

Los programas de la interfaz de usuario y los programas de aplicación pueden ejecutarse en el lado del cliente. Cuando se requiere acceso a DBMS, el programa establece una conexión al DBMS (que está en el lado del servidor); una vez que se crea la conexión, el cliente El programa puede comunicarse con el DBMS. Un estándar llamado **Open Database La conectividad** (**ODBC**) proporciona una **interfaz de programación de aplicaciones** (**API**), que permite que los programas del lado del cliente llamen al DBMS, siempre que tanto el cliente como las máquinas servidor tienen instalado el software necesario. La mayoría de los proveedores de DBMS vide controladores ODBC para sus sistemas. Un programa cliente realmente puede conectarse a varios RDBMS y enviar solicitudes de consulta y transacción utilizando la API ODBC, que luego se procesan en los sitios del servidor. Los resultados de la consulta se envían de nuevo al programa cliente, que puede procesar y mostrar los resultados según sea necesario. Un relacionado También se ha definido el estándar para el lenguaje de programación Java, llamado **JDBC** . Esto permite que los programas cliente de Java accedan a uno o más DBMS a través de un estándar. interfaz Dard.

Las arquitecturas descritas aquí se denominan **arquitecturas de dos niveles** porque el software Los componentes de software se distribuyen en dos sistemas: cliente y servidor. El avance Las características de esta arquitectura son su simplicidad y compatibilidad perfecta con sistemas. La aparición de la Web cambió los roles de clientes y servidores, liderando a la arquitectura de tres niveles.

2.5.4 Arquitecturas de tres y n niveles para aplicaciones web

Muchas aplicaciones web utilizan una arquitectura llamada arquitectura de **tres niveles** , que agrega una capa intermedia entre el cliente y el servidor de la base de datos, como ilustrado en la Figura 2.7 (a).

13 Existen muchas otras variaciones de arquitecturas cliente / servidor. Discutimos los dos más básicos aquí.

Página 50

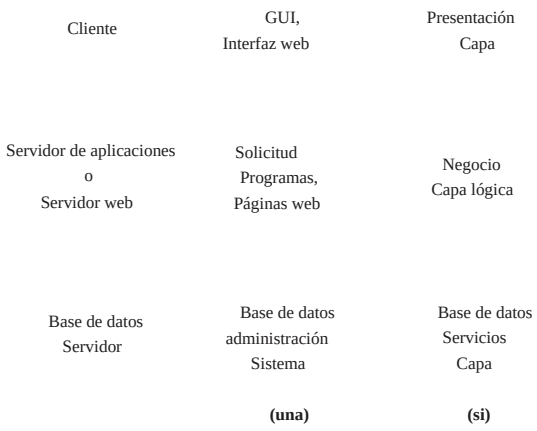


Figura 2.7
Lógico de tres niveles Servidor de cliente arquitectura, con un par de comúnmente nomenclaturas utilizadas.

Esta capa intermedia o **nivel medio** se denomina **servidor de aplicaciones** o **Web. servidor** , dependiendo de la aplicación. Este servidor juega un papel intermediario al ejecutar programas de aplicación y almacenar reglas comerciales (procedimientos o con- restricciones) que se utilizan para acceder a los datos del servidor de la base de datos. También puede mejorar seguridad de la base de datos comprobando las credenciales de un cliente antes de enviar una solicitud a el servidor de la base de datos. Los clientes contienen interfaces de usuario y navegadores web. El inter- El servidor de mediación acepta solicitudes del cliente, procesa la solicitud y envía consultas y comandos de la base de datos al servidor de la base de datos, y luego actúa como un conducto para pasar (parcialmente) datos procesados desde el servidor de la base de datos a los clientes, donde se puede procesar más y filtrar para presentarlo a los usuarios. Así, el usuario la interfaz, las reglas de la aplicación y el acceso a los datos actúan como los tres niveles. La figura 2.7 (b) muestra otra vista de la arquitectura de tres niveles utilizada por la base de datos y otras aplicaciones proveedores de paquetes. La capa de presentación muestra información al usuario y permite entrada de datos. La capa de lógica empresarial maneja las reglas y restricciones intermedias antes los datos se pasan al usuario o al DBMS. La capa inferior incluye todos servicios de gestión de datos. La capa intermedia también puede actuar como servidor web, que recupera los resultados de la consulta del servidor de la base de datos y los formatea en dinámicos Páginas web que el navegador web visualiza en el lado del cliente. La máquina cliente suele ser una PC o un dispositivo móvil conectado a la Web.

También se han propuesto otras arquitecturas. Es posible dividir las capas entre el usuario y los datos almacenados en componentes más finos, dando así se elevan a arquitecturas de n niveles, donde n puede ser de cuatro o cinco niveles. Normalmente, el negocio

La capa lógica se divide en varias capas. Además de distribuir programación y datos a través de una red, las aplicaciones de n niveles ofrecen la ventaja de que cualquier El nivel puede ejecutarse en un procesador apropiado o plataforma de sistema operativo y puede ser manejado de forma independiente. Proveedores de ERP (planificación de recursos empresariales) y CRM (gestión de relaciones con el cliente) los paquetes a menudo utilizan una capa de middleware, que

Página 51

cuentas para los módulos de front-end (clientes) que se comunican con una serie de bases de datos back-end (servidores).

Los avances en la tecnología de cifrado y descifrado hacen que sea más seguro transferir tive datos de servidor a cliente en forma cifrada, donde se descifrarán. los lo último se puede hacer por hardware o por software avanzado. Esta tecnología da niveles más altos de seguridad de datos, pero los problemas de seguridad de la red siguen siendo un problema importante cern. Varias tecnologías para la compresión de datos también ayudan a transferir grandes cantidades de datos de servidores a clientes a través de redes cableadas e inalámbricas.

2.6 Clasificación de la base de datos Sistemas de gestión

Se pueden utilizar varios criterios para clasificar los DBMS. El primero es el **modelo de datos** en en el que se basa el DBMS. El modelo de datos principal utilizado en muchos comerciales actuales DBMS es el **modelo de datos relacionales** , y los sistemas basados en este modelo son conocidos como **sistemas SQL.** El **modelo de datos de objetos** se ha implementado en algunos sistemas comerciales pero no ha tenido un uso generalizado. Recientemente, los llamados **big data sistemas,** también conocidos como **sistemas de almacenamiento de valores clave** y **los sistemas nosql,** uso variabilidad modelos de datos: **basados en documentos, basados en gráficos, basados en columnas** y **clave-valor modelos de datos.** Muchas aplicaciones heredadas todavía se ejecutan en sistemas de bases de datos basados en **modelos de datos jerárquicos** y de **red** .

Los DBMS relacionales evolucionan continuamente y, en particular, se han incorporando muchos de los conceptos que se desarrollaron en bases de datos de objetos. Esta ha dado lugar a una nueva clase de DBMS denominados **DBMS relacionales de objetos** . Podemos categorize DBMS basados en el modelo de datos: relacional, objeto, objeto-relacional, NOSQL, valor-clave, jerárquico, de red y otros.

Algunos DBMS experimentales se basan en XML (eXtended Markup Language) modelo, que es un **modelo de datos estructurado en árbol.** Estos se han llamado **XML nativo DBMS.** Varios DBMS relacionales comerciales han agregado interfaces XML y almacenamiento a sus productos.

El segundo criterio utilizado para clasificar los DBMS es el **número de usuarios** admitidos por el sistema. **Los sistemas de** un solo **usuario** admiten solo un usuario a la vez y en su mayoría son utilizado con PC. **Los sistemas multiusuario** , que incluyen la mayoría de DBMS, admiten múltiples usuarios concurrentes.

El tercer criterio es el **número de sitios** en los que se distribuye la base de datos. UNA

El DBMS está **centralizado** si los datos se almacenan en un solo sitio de computadora. Centralizado DBMS puede soportar múltiples usuarios, pero el DBMS y la base de datos residen totalmente en un solo sitio informático. Un DBMS **distribuido** (DDBMS) puede tener la base de datos real y software DBMS distribuido en muchos sitios conectados por una red informática. Los grandes sistemas de datos a menudo se distribuyen masivamente, con cientos de sitios. Los datos son a menudo se replica en varios sitios para que la falla de un sitio no genere algunos datos indisponible.

Página 52

Los DDBMS **homogéneos** utilizan el mismo software DBMS en todos los sitios, mientras que **Los** DDBMS **heterogéneos** pueden utilizar diferentes software DBMS en cada sitio. Tambien es posible desarrollar **software de middleware** para acceder a varios preexistentes autónomos bases de datos almacenadas en DBMS heterogéneos. Esto conduce a un DBMS **federado** (o **sistema de base de datos múltiple**), en el que los DBMS participantes están débilmente acoplados y tener cierto grado de autonomía local. Muchos DDBMS utilizan la arquitectura cliente-servidor, como que describimos en la Sección 2.5.

El cuarto criterio es el costo. Es difícil proponer una clasificación de DBMS basado en el costo. Hoy tenemos productos DBMS de código abierto (gratuitos) como MySQL y PostgreSQL que son compatibles con proveedores externos con servicios adicionales. Los principales productos RDBMS están disponibles como versiones de copia de 30 días de examen gratuito así como versiones personales, que pueden costar menos de $ 100 y permitir una buena cantidad de funcionalidad. Los sistemas gigantes se venden en forma modular con componentes para manejar la distribución, la replicación, el procesamiento paralelo, la capacidad móvil, etc. encendido, y con una gran cantidad de parámetros que deben definirse para la configuración ción. Además, se venden en forma de licencias; las licencias de sitio permiten Uso ordenado del sistema de base de datos con cualquier número de copias ejecutándose en el cliente. sitio. Otro tipo de licencia limita el número de usuarios concurrentes o el número de asientos de usuario en una ubicación. Versiones independientes para un solo usuario de algunos sistemas como Microsoft Access se vende por copia o se incluye en la configuración general de un computadora de escritorio o portátil. Además, las funciones de almacenamiento de datos y minería, así como soporte para tipos de datos adicionales, están disponibles a un costo adicional. Es posible Pagar millones de dólares por la instalación y el mantenimiento de grandes sistemas de bases de datos. tems anualmente.

También podemos clasificar un DBMS en función de los **tipos de** opciones de **ruta de acceso** para almacenar archivos. Una familia conocida de DBMS se basa en estructuras de archivos invertidas. Finalmente, un DBMS puede ser de **propósito general** o **especial** . Cuando el rendimiento es Una consideración primordial, un DBMS de propósito especial se puede diseñar y construir para un aplicación específica; tal sistema no se puede utilizar para otras aplicaciones sin cambios principales. Muchas reservas de aerolíneas y sistemas de directorio telefónico se desarrollan utilizados en el pasado son DBMS para propósitos especiales. Estos entran en la categoría de **online** sistemas de **procesamiento de transacciones** (**OLTP**), que deben admitir una gran cantidad de transacciones concurrentes sin imponer demoras excesivas.

Analicemos brevemente el criterio principal para clasificar los DBMS: los datos modelo. El **modelo de datos relacionales** representa una base de datos como una colección de tablas, donde cada tabla se puede almacenar como un archivo separado. La base de datos de la Figura 1.2 parece

bles una representación relacional básica. La mayoría de las bases de datos relacionales utilizan el lenguaje de consulta llamado SQL y admite una forma limitada de vistas de usuario. Discutimos el modelo relacional y sus lenguajes y operaciones en los Capítulos 5 al 8, y técnicas para programar aplicaciones relacionales en los capítulos 10 y 11.

El **modelo de datos de objetos** define una base de datos en términos de objetos, sus propiedades y sus operaciones. Los objetos con la misma estructura y comportamiento pertenecen a una **clase** , y las clases se organizan en **jerarquías** (o **gráficos acíclicos**). Las operaciones de

Página 53

cada clase se especifica en términos de procedimientos predefinidos llamados **métodos** . Rela-
Los DBMS tradicionales han ampliado sus modelos para incorporar bases de datos de objetos.
conceptos y otras capacidades; estos sistemas se conocen como **relacionales de objetos** o
sistemas relacionales extendidos . Discutimos bases de datos de objetos y objetos relacionales
sistemas en el Capítulo 12.

Los sistemas de big data se basan en varios modelos de datos, con los siguientes cuatro datos
modelos más comunes. El **modelo de datos clave-valor** asocia una clave única con
cada valor (que puede ser un registro u objeto) y proporciona un acceso muy rápido a un
valor dado su clave. El **modelo de datos del documento** se basa en JSON (Java Script
Notación de objetos) y almacena los datos como documentos, que se parecen un poco
objetos complejos. El **modelo de datos de gráficos** almacena objetos como nodos de gráficos y
relaciones entre objetos como bordes de gráficos dirigidos. Finalmente, los **datos basados en columnas**
Los modelos almacenan las columnas de filas agrupadas en páginas de disco para un acceso rápido y
permitir múltiples versiones de los datos. Discutiremos algunos de estos con más detalle.
en el Capítulo 24.

El **modelo XML** ha surgido como un estándar para el intercambio de datos en la Web y
se ha utilizado como base para implementar varios prototipos de sistemas XML nativos.
XML utiliza estructuras de árbol jerárquicas. Combina conceptos de bases de datos con conceptos
a partir de modelos de representación de documentos. Los datos se representan como elementos; con el
uso de etiquetas, los datos se pueden anidar para crear estructuras de árbol complejas. Este modelo
ceptualmente se parece al modelo de objetos pero utiliza una terminología diferente. Capacidad XML
Se han agregado vínculos a muchos productos DBMS comerciales. Presentamos una descripción general
de XML en el Capítulo 13.

Dos modelos de datos más antiguos e históricamente importantes, ahora conocidos como **modelos de datos heredados** ,
son la red y los modelos jerárquicos. El **modelo de red** representa los datos como
tipos de registro y también representa un tipo limitado de relación 1: N, llamado **tipo de conjunto** .
Una relación 1: N, o uno a muchos, relaciona una instancia de un registro con muchos registros
instancias que utilizan algún mecanismo de enlace de puntero en estos modelos. La red
modelo, también conocido como el modelo CODASYL DBTG, [14] tiene un registro asociado
un lenguaje de tiempo que debe estar integrado en un lenguaje de programación de host. La red-
trabajo DML se propuso en el Informe del Grupo de Trabajo de Base de Datos (DBTG) de 1971 como un
extensión del lenguaje COBOL.

El **modelo jerárquico** representa los datos como estructuras de árbol jerárquicas. Cada jerarquía
chy representa varios registros relacionados. No existe un idioma estándar para

modelo jerárquico. Un DML jerárquico popular es DL / 1 del sistema IMS. Domina en el mercado de DBMS durante más de 20 años entre 1965 y 1985. Su DML, llamado DL / 1, fue un estándar industrial de facto durante mucho tiempo. [15]

[14] CODASYL DBTG son las siglas de Conference on Data Systems Languages Database Task Group, que es el comité que especificó el modelo de red y su idioma.

[15] Los capítulos completos sobre la red y los modelos jerárquicos de la segunda edición de este libro son disponible en el sitio Web Companion de este libro en http://www.aw.com/elmasri.

Página 54

2.7 Resumen

En este capítulo presentamos los principales conceptos utilizados en los sistemas de bases de datos. Nosotros definimos un modelo de datos y distinguimos tres categorías principales:

- Modelos de datos conceptuales o de alto nivel (basados en entidades y relaciones)
- Modelos de datos físicos o de bajo nivel
- Modelos de datos representativos o de implementación (basados en registros, objetos orientado)

Distinguimos el esquema, o descripción de una base de datos, de la propia base de datos. El esquema no cambia muy a menudo, mientras que el estado de la base de datos cambia cada hora en que se insertan, eliminan o modifican los datos. Luego describimos el esquema de tres Arquitectura DBMS, que permite tres niveles de esquema:

- Un esquema interno describe la estructura de almacenamiento físico de la base de datos.
- Un esquema conceptual es una descripción de alto nivel de toda la base de datos.
- Los esquemas externos describen las vistas de diferentes grupos de usuarios.

Un DBMS que separa limpiamente los tres niveles debe tener asignaciones entre los esquemas para transformar las solicitudes y los resultados de las consultas de un nivel al siguiente. La mayoría de los DBMS no separan completamente los tres niveles. Usamos el Arquitectura de tres esquemas para definir los conceptos de datos lógicos y físicos. independencia.

Luego discutimos los principales tipos de lenguajes e interfaces que admiten los DBMS. Se utiliza un lenguaje de definición de datos (DDL) para definir el esquema conceptual de la base de datos. En la mayoría de los DBMS, el DDL también define las vistas del usuario y, a veces, la estructura de almacenamiento. tures; en otros DBMS, existen lenguajes o funciones separados para especificar el almacenamiento estructuras. Esta distinción se está desvaneciendo en las implementaciones relacionales de hoy, con SQL sirviendo como un lenguaje general para realizar múltiples funciones, incluida la vista definición. La parte de definición de almacenamiento (SDL) se incluyó en las primeras versiones de SQL, pero ahora normalmente se implementa como comandos especiales para el DBA en relaciones DBMS. El DBMS compila todas las definiciones de esquema y almacena sus descripciones en el catálogo DBMS.

Se utiliza un lenguaje de manipulación de datos (DML) para especificar recuperaciones de bases de datos y actualizaciones. Los DML pueden ser de alto nivel (orientados a conjuntos, no procedimentales) o de bajo nivel (registro orientado, procedimental). Se puede incrustar un DML de alto nivel en una programación de host idioma, o puede usarse como un idioma independiente; en el último caso es a menudo llamado lenguaje de consulta.

Discutimos diferentes tipos de interfaces proporcionados por DBMS y los tipos de Usuarios de DBMS a los que se asocia cada interfaz. Luego discutimos el entorno del sistema de base de datos, módulos de software DBMS típicos y DBMS utilidades para ayudar a los usuarios y al personal de DBA a realizar sus tareas. Continuamos con una descripción general de las arquitecturas de dos y tres niveles para la base de datos aplicaciones.

Página 55

Finalmente, clasificamos los DBMS de acuerdo con varios criterios: modelo de datos, número de usuarios, número de sitios, tipos de rutas de acceso y costo. Discutimos la disponibilidad de DBMS y módulos adicionales, sin costo en forma de código abierto software a configuraciones cuyo mantenimiento cuesta anualmente millones. Nosotros también señaló la variedad de acuerdos de licencia para DBMS y productos relacionados ucts. La clasificación principal de los DBMS se basa en el modelo de datos. Brevemente discutió los principales modelos de datos utilizados en los DBMS comerciales actuales.

Preguntas de revisión

2.1. Defina los siguientes términos: modelo de datos, esquema de la base de datos, estado de la base de datos, esquema interno, esquema conceptual, esquema externo, independencia de datos, DDL, DML, SDL, VDL, lenguaje de consulta, lenguaje host, sublenguaje de datos, utilidad de base de datos, catálogo, arquitectura cliente / servidor, arquitectura de tres niveles, y arquitectura de n niveles.

2.2. Analice las principales categorías de modelos de datos. Cuales son las diferencias basicas entre el modelo relacional, el modelo de objetos y el modelo XML?

2.3. ¿Cuál es la diferencia entre un esquema de base de datos y un estado de base de datos?

2.4. Describe la arquitectura de tres esquemas. ¿Por qué necesitamos mapeos entre niveles de esquema? ¿Cómo admiten esto los diferentes lenguajes de definición de esquemas? ¿arquitectura?

2.5. ¿Cuál es la diferencia entre la independencia de datos lógicos y los datos físicos? ¿independencia? ¿Cuál es más difícil de lograr? ¿Por qué?

2.6. ¿Cuál es la diferencia entre LMD procedimentales y no procedimentales?

2.7. Discutir los diferentes tipos de interfaces fáciles de usar y los tipos de usuarios que suelen utilizar cada uno.

2.8. ¿Con qué otro software de sistema informático interactúa un DBMS?

2.9. ¿Cuál es la diferencia entre el cliente / servidor de dos y tres niveles?

arquitecturas?

2.10. Analice algunos tipos de utilidades y herramientas de bases de datos y sus funciones.

2.11. ¿Cuál es la funcionalidad adicional incorporada en la arquitectura de n niveles? (n. 3)?

Ejercicios

2.12. Piense en diferentes usuarios para la base de datos que se muestra en la Figura 1.2. ¿Qué tipos de aplicaciones que necesitaría cada usuario? ¿A qué categoría de usuario cada uno pertenecen, y qué tipo de interfaz necesitaría cada uno?

Página 56

2.13. Elija una aplicación de base de datos con la que esté familiarizado. Diseñar un esquema y mostrar una base de datos de muestra para esa aplicación, utilizando la notación de la figura ures 1.2 y 2.1. Qué tipos de información adicional y limitaciones le gustaría representar en el esquema? Piense en varios usuarios de su base de datos y diseñar una vista para cada uno.

2.14. Si estuviera diseñando un sistema basado en la web para hacer reservas de aerolíneas y vender billetes de avión, ¿qué arquitectura DBMS elegiría de la Sección 2.5? ¿Por qué? ¿Por qué las otras arquitecturas no serían una buena opción?

2.15. Considere la Figura 2.1. Además de las restricciones que relacionan los valores de columns en una tabla a columnas en otra tabla, también hay restricciones que imponer restricciones sobre los valores en una columna o una combinación de columnas dentro de una mesa. Una de esas restricciones dicta que una columna o un grupo de Las umns deben ser únicas en todas las filas de la tabla. Por ejemplo, en el ESTUDIANTE , la columna Número_alumno debe ser única (para evitar dos estudiantes diferentes de tener el mismo número de estudiante). Identificar la columna o el grupo de columnas en las otras tablas que deben ser únicas en todas las filas de la tabla.

Bibliografía seleccionada

Muchos libros de texto de bases de datos, incluido Date (2004), Silberschatz et al. (2011), Ramakrishnan y Gehrke (2003), García-Molina et al. (2002, 2009) y Abiteboul et al. (1995), proporcionan una discusión de los diversos conceptos de bases de datos presentados aquí. Tsichritzis y Lochovsky (1982) es uno de los primeros libros de texto sobre modelos de datos. Tsichritzis y Klug (1978) y Jardine (1977) presentan la arquitectura de tres esquemas, que fue sugerido por primera vez en el informe DBTG CODASYL (1971) y más tarde en un Informe del Instituto Nacional de Normas (ANSI) (1975). Un análisis en profundidad de las relaciones El modelo de datos nacionales y algunas de sus posibles extensiones se dan en Codd (1990). los El estándar propuesto para bases de datos orientadas a objetos se describe en Cattell et al. (2000).

Muchos documentos que describen XML están disponibles en la Web, como XML (2005).
Ejemplos de utilidades de bases de datos son las herramientas ETI Connect, Analyze y Transform (http://www.eti.com) y la herramienta de administración de bases de datos, DBArtisan, de Embarcadero Technologies (http://www.embarcadero.com).