UNIVERSIDAD DE LA EMPRESA

CAPITULO 1
INTRODUCCION A LOS SISTEMAS DE BASES DE DATOS

En los Sistemas de Información uno de los nuevos elementos, que se ha transformado en uno de los más importantes, son los Sistemas Manejadores de Bases de Datos.

Este material es de uso exclusivo para los cursos impartidos por Universidad de la Empresa y asociados

1.1 INTRODUCCION

1.1.1 Sistemas de archivos vs. sistemas de bases de datos

Considérese una empresa, por ejemplo una aerolínea, que almacena una gran cantidad de datos en una computadora, durante mucho tiempo. Estos datos incluyen información sobre los pasajeros, los vuelos, los aviones y el personal de la aerolínea. También se incluye información sobre relaciones entre los datos, como son las reservas de pasajes, las tripulaciones de vuelos y los servicios de mantenimiento realizados a los aviones.

En una empresa típica, los registros referentes a las aplicaciones se mantienen en archivos. Además de estos archivos, el sistema cuenta con programas de aplicación que permiten manejar los archivos de datos. Cada vez que se necesita, se agregan nuevos programas al sistema. Como resultado, se crean nuevos archivos que contienen toda la información requerida. Es decir, con el correr del tiempo se agregan al sistema, archivos y programas. Puesto que tales archivos y programas se han creado en un período largo y, probablemente, por distintos programadores, es de esperar que los archivos tengan formatos diferentes y que los programas estén escritos en varios lenguajes de programación.

El ambiente que se acaba de describir es un sistema de procesamiento de archivos clásico, apoyado por un sistema operativo convencional. Los registros permanentes se guardan en diversos archivos, y se escriben varios programas para sacar registros y agregarlos a los archivos apropiados. Este sistema tiene ciertas desventajas importantes:

- *Redundancia e inconsistencia de datos:* Puesto que los archivos y los programas fueron creados por distintos programadores en un período largo, es posible que un mismo dato esté repetido en varios sitios (archivos). Esta redundancia aumenta los costos de almacenamiento y acceso, además de incrementar la posibilidad de que exista inconsistencia en la información, es decir, que las distintas copias de la misma información no concuerden entre sí.

- *Dificultad para tener acceso a los datos:* Supóngase que uno de los gerentes de la aerolínea necesita averiguar los nombres de las azafatas que viven en cierta parte de la ciudad. El gerente llama al departamento de procesamiento de datos y pide que generen la lista correspondiente. Como ésta es una solicitud fuera de lo común, que no estaba prevista cuando se diseñó el sistema original, no existe un programa de aplicaciones para generar semejante lista. Lo que sí existe es un programa de aplicaciones que genera una lista de *todos* los empleados que viven en una cierta parte de la ciudad. El gerente o bien solicita la lista de todos y se encarga de buscar manualmente la información, o solicita que uno de los programadores escriba el programa correspondiente. Ninguna de las dos opciones es satisfactoria. Lo que se trata de mostrar aquí es que este ambiente no permite recuperar información en forma conveniente o eficiente.

- *Aislamiento de los datos:* Puesto que los datos están repartidos en varios archivos, y éstos pueden tener diferentes formatos, es difícil escribir nuevos programas para obtener los datos apropiados.

Este material es de uso exclusivo para los cursos impartidos por Universidad de la Empresa y asociados

- *Usuarios múltiples:* Para mejorar el funcionamiento general del sistema y tener un tiempo de respuesta más corto, muchos sistemas permiten que varios usuarios actualicen la información en forma simultánea. En un ambiente de este tipo, la interacción de las actualizaciones concurrentes puede resultar en información inconsistente. Para prevenir estas situaciones debe mantenerse alguna forma de supervisión en el sistema. Puesto que muchos programas diferentes sin coordinación previa pueden tener acceso a los datos, es muy difícil de conseguir un supervisor de este tipo.

- *Problemas de seguridad:* No es recomendable que todos los usuarios del sistema de base de datos puedan tener acceso a *toda* la información. Puesto que los programas de aplicaciones se agregan al sistema en una forma precisa, es difícil implantar estas limitantes de seguridad.

- *Problemas de integridad:* Los valores de los datos que se guardan en la base de datos deben satisfacer ciertos tipos de *restricciones de integridad* o *limitantes de consistencia.* Por ejemplo, la cantidad de asientos reservados de un vuelo no debe pasar nunca de un límite fijado (por ejemplo, la capacidad de asientos del avión asignado a dicho vuelo). El sistema debe obligar al cumplimiento de estas restricciones. Esto puede hacerse agregando el código de programa apropiado. Sin embargo, cuando se añaden restricciones nuevas, es difícil cambiar los programas para asegurar su cumplimiento. El problema se complica aún más cuando las restricciones implican varios elementos de información de distintos archivos.

Estos problemas, entre otros, han fomentado el desarrollo de los sistemas de manejo de bases de datos.

Este material es de uso exclusivo para los cursos impartidos por Universidad de la Empresa y asociados

1.1.2 Estructura general de un DBMS

Los sistemas de bases de datos se diseñan para manejar grandes cantidades de información. El manejo de los datos incluye tanto la definición de las estructuras para el almacenamiento de la información como los mecanismos para el manejo de la misma. Además, el sistema de base de datos debe cuidar la seguridad de la información almacenada, tanto contra las caídas del sistema como contra los intentos de acceso no autorizado. Si los datos van a ser compartidos por varios usuarios, el sistema debe evitar la posibilidad de obtener resultados anómalos.

Un *sistema de bases de datos* consiste en una base de datos, un sistema manejador de la base de datos y un conjunto de programas de los usuarios. Una *base de datos* es un depósito de datos almacenados en dispositivos de memoria secundaria (discos). El *sistema manejador de la base de datos* es el software cuyas funciones principales son dar acceso de alto nivel a los datos almacenados y mantener la consistencia e integridad de los mismos. Los *programas de usuarios* expresan los tipos de consultas y modificaciones que los usuarios desean realizar sobre la base de datos. Tales programas acceden a la base de datos utilizando los servicios ofrecidos por el sistema manejador de la base de datos.

Denominamos *base de datos* a los datos tales como los mencionados antes, y que se almacenan más o menos permanentemente en un computador. El software que permite a una o varias personas usar y/o modificar estos datos se denomina *sistema manejador de la base de datos (Data Base Management System, DBMS).* El mayor rol de un DBMS es permitirle al usuario tratar con los datos en términos abstractos, en vez de tratarlos en la forma tal como la computadora los almacena. En este sentido, el DBMS actúa como un intérprete de un lenguaje de programación de alto nivel, permitiendo al usuario especificar lo que se debe hacer, con poca o ninguna atención de parte del usuario a los algoritmos detallados o a la representación de datos usada en el sistema.

Mostraremos la estructura general de un sistema de base de datos, dividida en módulos, cada uno de los cuales se encarga de una de las responsabilidades del sistema total. El sistema operativo de la computadora realiza algunas de las funciones del sistema de base de datos, pero en la mayor parte de los casos el sistema operativo proporciona únicamente los servicios más elementales, y el sistema de bases de datos debe construir sobre esos cimientos. Por tanto, al hablar del diseño de una base de datos debe tomarse en consideración la interfaz entre el sistema de base de datos y el sistema operativo. Un DBMS consta de varias *componentes funcionales* o *módulos,* entre los que se encuentran:

- *El manejador de archivos,* que se encarga de asignar espacio físico dentro de los discos y de las estructuras de datos que se emplean para representar la información almacenada en él.

- *El manejador de buffer,* que se encarga de transferir información entre el disco y la memoria principal.

- El *procesador de consultas* se encarga de manejar las sentencias del *lenguaje de consulta* que se utiliza para acceder a los datos. Consta de 3 partes: *El analizador sintáctico (parser) de consultas,* que traduce las sentencias de un lenguaje de consulta a uno de más bajo nivel; el optimizador de consultas, que procura transformar la solicitud del usuario en una forma equivalente pero más eficiente; y el *selector de estrategias* que se encarga de encontrar una estrategia adecuada para ejecutar la consulta.

Este material es de uso exclusivo para los cursos impartidos por Universidad de la Empresa y asociados

- *El manejador de recuperaciones,* que se asegura de que la base de datos permanezca en un estado correcto a pesar de que ocurran fallas en el sistema.

- *El controlador de concurrencia,* que garantiza que las interacciones concurrentes con la base de datos se lleven a cabo sin conflictos entre ellas.

Además se requieren varias *estructuras de datos* como parte de la implementación física del sistema:

- *Archivos de datos,* en los que se almacenan los datos.

- *Diccionario de datos del sistema,* que guarda información acerca de la estructura de la base de datos. Aquí se conserva la información de autorizaciones de acceso, claves de búsqueda de datos, etc.

- *Índices,* que permiten acceso rápido a datos que tienen ciertos valores.

- *Información estadística,* que incluye información acerca de los datos almacenados en la base de datos. Esta información la utiliza el selector de estrategias.

1.1.3 Los usuarios de un DBMS

Los DBMSs son una variedad de software complejo. Una forma de percibir los diferentes aspectos de un DBMS es considerar los tipos diferentes de usuarios de tales sistemas y la forma en la que interactúan con el mismo.

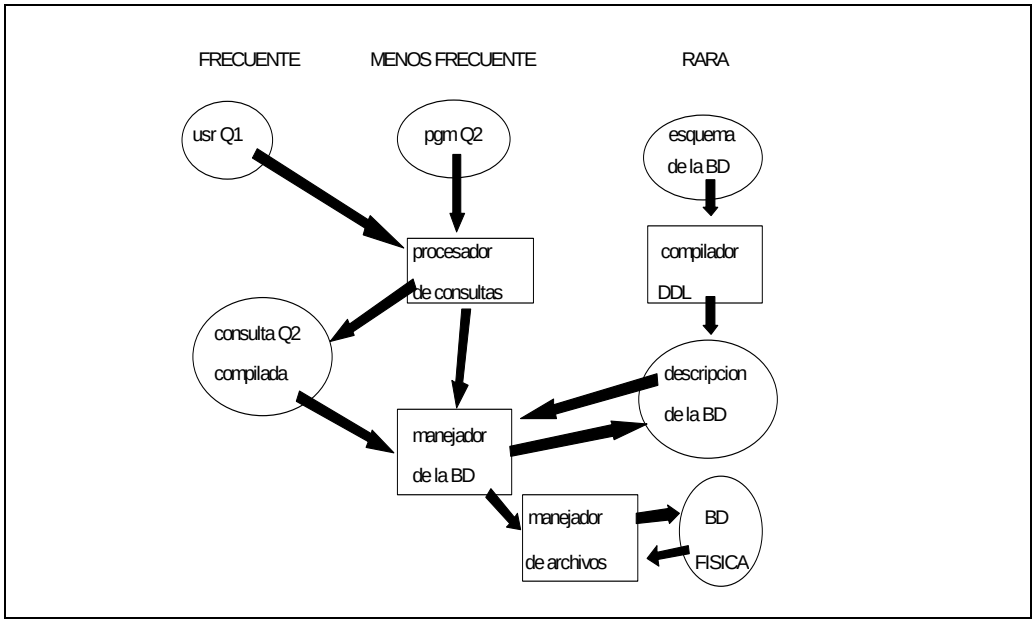
- **El programador/usuario**

Ilustr. 2 El usuario programador.

Volviendo a la base de datos de la aerolínea, un usuario con alguna capacitación como programador desea averiguar cuáles aviones están en reparación y dónde están ubicados. El usuario puede formular su consulta usando algún *lenguaje de consulta (Query Language, QL).*

La ilustración muestra qué ocurre con dicha consulta, representada como *Q1.*

Primero es tomada por el procesador de consultas, que es como un compilador de la consulta, aunque la salida no es código de máquina sino que es una secuencia de comandos que son pasados a otras partes del DBMS. El procesador de consultas necesita saber la estructura de la base de datos, accediendo a la *descripción de la base de datos.* Esta información es necesaria a los efectos de que se puedan interpretar los términos "Avion", "Ubicacion" y "Estado" dentro del contexto de esta base de datos en particular. La descripción de la base de datos está escrita en un lenguaje especializado denominado *lenguaje de definición de datos (Data Definition Language, DDL)* que está compilado en tablas que son usadas por el resto del DBMS.



Ilustr. 3 Diagrama esquematico de un sistema de base de datos.

La consulta procesada es pasada a una colección de rutinas a las que denominaremos *manejador de la base de datos.* Un rol de este manejador es traducir la consulta a términos que puedan ser comprendidos por el *manejador de archivos,* es decir, operaciones sobre archivos, en vez de estructuras de datos más abstractas. Este manejador de archivos puede ser el sistema de archivos provisto por el sistema operativo, o puede ser un sistema de archivos especializado que conoce la forma particular en la que los datos estan almacenados en la base de datos. La traducción de la consulta a operaciones sobre archivos no es trivial debido a que la base de datos puede estar representada por estructuras de archivos complejas, cuyo propósito es acceder y manipular la base de datos tan rápido como sea posible.

El manejador de base de datos frecuentemente ejecuta otras tareas:

1. *Seguridad:* No todos los usuarios tienen acceso a todos los datos. Por ejemplo, si se mantienen fichas del personal, solamente el personal adecuado puede acceder a los

Este material es de uso exclusivo para los cursos impartidos por Universidad de la Empresa y asociados

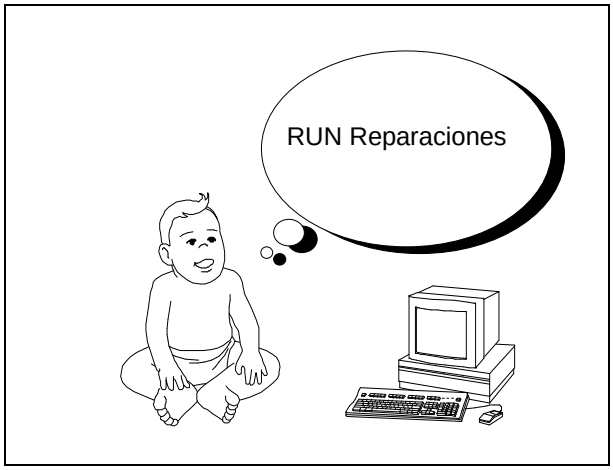
datos de salarios. El usuario adecuado presumiblemente se ha identificado mediante una contraseña y sabe que tiene derechos para acceder a los datos de salarios.

2. Integridad: Si se le requiere, el DBMS puede controlar algunas *restricciones de integridad* (es decir, propiedades requeridas sobre los datos). Es útil tener tales controles cada vez que un usuario da un comando usando el *lenguaje de manipulación de datos (Data Manipulation Language, DML)* para insertar, borrar o modificar datos. Las propiedades más fáciles de controlar son las propiedades de los valores, por ejemplo que la cantidad de pasajeros que reservaron en un vuelo, no exceda la capacidad del avión. Otras restricciones son más difíciles de controlar, por ejemplo que dos aviones no estén asignados al mismo vuelo.

3. Sincronización: A menudo varios usuarios están corriendo programas que acceden al mismo tiempo a la base de datos. El DBMS debe proveer protección contra inconsistencias que pudieran resultar de dos operaciones aproximadamente simultáneas sobre los mismos datos. Por ejemplo, supóngase de que más o menos al mismo tiempo, se hacen dos reservaciones para un asiento en el vuelo 999. Cada reservación provoca la ejecución de un programa que examina el numero de asientos disponibles (supongamos que queda 1), decrementa en uno la cantidad disponible y almacena el nuevo valor en la base de datos. Si el DBMS no establece una secuenciación de las dos transacciones (las dos invocaciones al programa de reservas), dos pasajeros deberán sentarse en el mismo asiento!

■ **El usuario inexperto**

Otro usuario sin la pericia del anterior, podría querer hacer la misma consulta de aviones en reparación, pero simplemente tecleando *"run Reparaciones."*



Ilustr. 4 El usuario inexperto.

En forma similar, los empleados en las ventanillas de reservas ejecutarían en sus terminales el programa de reservas. El programa invocado establece un diálogo con el usuario, pidiéndole información, por ejemplo "ingrese el nombre del pasajero" e "ingrese el numero de vuelo deseado". Una vez obtenida esta información, el programa interroga a la base de datos para determinar si hay asientos disponibles, y si los hay, modifica la base de datos. En caso negativo, el programa informa al usuario.

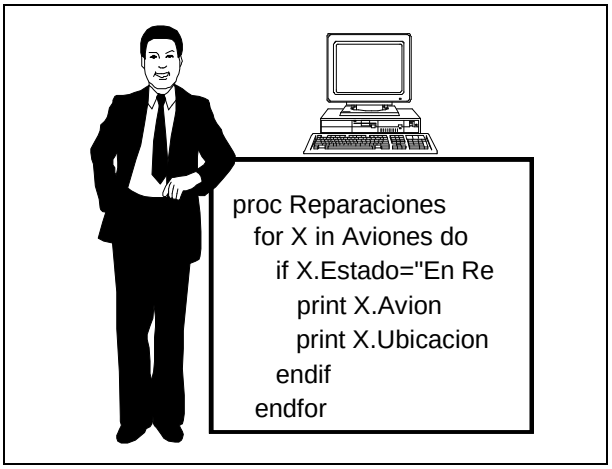
■ **El programador de aplicaciones**

Este material es de uso exclusivo para los cursos impartidos por Universidad de la Empresa y asociados

Los programas (por ejemplo Reparaciones o Reservas) almacenados y disponibles a los usuarios se denominan *programas de aplicación.* Estos programas fueron escritos por un programador de aplicaciones usando algun lenguaje de manipulación de datos. Estos programas fueron compilados y almacenados en el sistema de archivos.

■ El administrador de la base de datos

En algunos -raros- casos la descripción de la base de datos es cambiada. Es decir, el programa en lenguaje de definición de datos que describe a la base de datos es modificado y recompilado. Esta operación infrecuente pero importante es responsabilidad del usuario *administrador de la base de datos,* cuyas responsabilidades son:



Ilustr. 5 El usuario programador de aplicaciones.

1. La creación de la descripción original de la estructura de la base de datos y la forma en la que la estructura es reflejada por los archivos de la base de datos física.

2. La otorgación de las autorizaciones de acceso a los demás usuarios.

3. La modificación de la descripción de la base de datos o su relación con la organización física, cuando la experiencia indica que otra organización física es más eficiente (por ejemplo, generar un archivo de índice para una colección de datos que ha crecido demasiado).

4. La realización de copias de respaldo de la base de datos y la reparación de errores debido a fallas de hardware o de software.

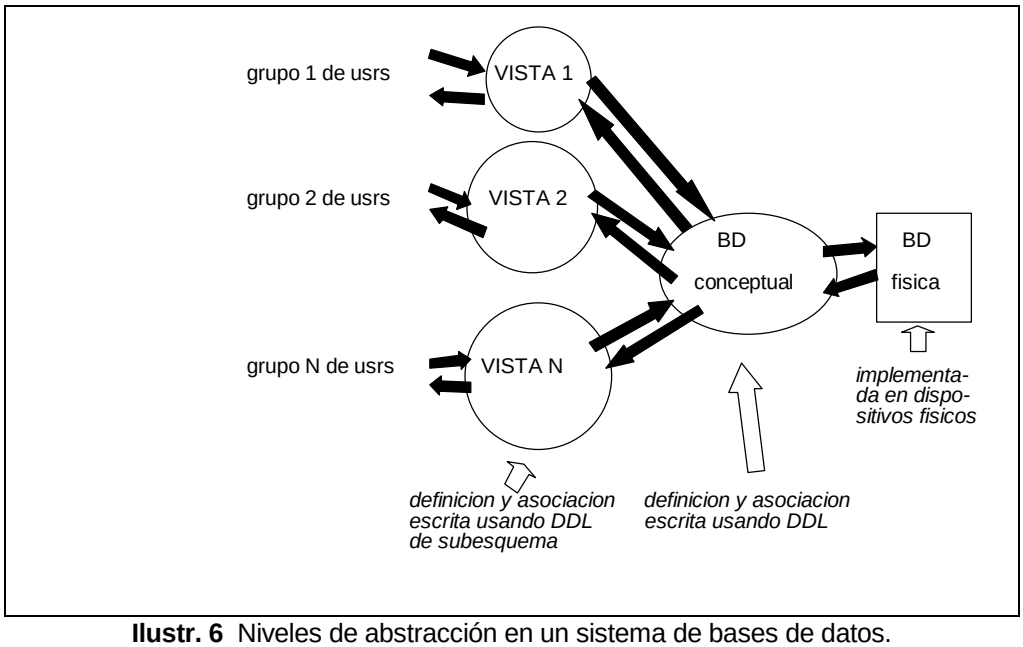
Este material es de uso exclusivo para los cursos impartidos por Universidad de la Empresa y asociados

1.2 TERMINOLOGIA DE UN DBMS

En esta sección desarrollaremos los conceptos ya introducidos en forma más precisa. Hay varios tipos de distinciones para hacer cuando se habla de un sistema de bases de datos. Comenzaremos por discutir los tres niveles de abstracción usados al describir bases de datos. Enfatizaremos la dicotomía *esquema/instancia,* es decir, la distinción entre los planes de una cosa y la cosa en sí misma. Finalmente discutiremos los diferentes tipos de lenguajes usados en un sistema de bases de datos.

1.2.1 Niveles de abstracción

Resulta obvio que entre la computadora, que trata con bits, y el usuario final, que trata con abstracciones como vuelos o asignación de personal a los aviones, deben haber varios niveles de abstracción. Un punto de vista bastante estándar respecto a los niveles de abstracción se muestra en la ilustración:



Ilustr. 6 Niveles de abstracción en un sistema de bases de datos.

Vemos que hay una única base de datos, que puede ser una de varias bases de datos usando el mismo software DBMS, a tres niveles de abstracción. Debemos resaltar que realmente sólo existe la base de datos física.

El *nivel físico* consiste en la base de datos que reside permanentemente en dispositivos de almacenamientos secundario, como son los discos. Podemos ver a la base de datos física desde

Este material es de uso exclusivo para los cursos impartidos por Universidad de la Empresa y asociados

varios niveles de abstracción, desde los registros y archivos de un lenguaje de programación como *Pascal,* pasando por el nivel de registros lógicos soportados por el sistema operativo bajo el cual corre el DBMS, hasta el nivel de bits y direcciones físicas de los dispositivos de almacenamiento.

El *nivel conceptual* es una abstracción del mundo real, al nivel de pasajeros, vuelos, etc. Un DBMS provee un lenguaje de definición de datos para especificar el esquema conceptual y dar algunos detalles referentes a la implementación del esquema conceptual por parte del esquema físico. El lenguaje de definición de datos es un lenguaje de alto nivel que nos permite describir la base de datos conceptual en términos de un *modelo de datos.*

El *nivel externo o de vista* es un modelo abstracto de una porción de la base de datos conceptual. Muchos, pero no todos los DBMSs, tienen facilidades para declarar vistas, expresar consultas y hacer operaciones sobre las mismas.

Como ejemplo de la utilidad de las vistas, una aerolínea puede proveer un servicio de reservas computarizado, consistente en los datos y los programas que tratan con vuelos y pasajeros. Estos programas y la gente que los usa, no necesitan saber sobre los datos del personal o la asignación de pilotos a vuelos. A su vez, los despachadores de vuelo deben saber los datos de vuelos, aviones y personal apto para ellos. Pero no necesitan saber sobre los salarios del personal o sobre los pasajeros que han reservado en los vuelos. En otras palabras, hay una vista de la base de datos para el personal de reservas y otra vista para el personal despachador de vuelos.

En cierto sentido, una vista es una base de datos conceptual pequeña y está al mismo nivel de abstracción de la base conceptual. Sin embargo, en cierto sentido una vista puede ser "más abstracta" que una base de datos conceptual, ya que los datos de la vista se construyen de la base de datos conceptual pero no están realmente presentes en la base de datos.

Como ejemplo, el departamento de personal puede tener una vista que incluya la edad de cada empleado. Sin embargo es improbable que las edades estén en la base de datos conceptual, ya que las mismas deberían ser cambiadas a diario para algunos empleados. Más bien la base de datos conceptual incluye la fecha de nacimiento de cada empleado. Cuando un programa solicita a la vista la edad del empleado, el DBMS traduce este requerimiento en "fecha actual menos fecha de nacimiento," que tiene sentido para la base de datos conceptual, y el cálculo puede hacerse sobre los datos correspondientes tomados de la base de datos física.

1.2.2 Esquemas e instancias

Además de los distintos grados de niveles de abstracción hay otra dimensión ortogonal a nuestra percepción de las bases de datos. Cuando la base de datos es diseñada, estamos interesados en planes para la misma, mientras que cuando la usamos, estamos interesados en los datos reales presentes. Nótese que los datos en la base de datos cambian con frecuencia, mientras que los planes permanecen por períodos largos de tiempo (aunque no necesariamente para siempre).

Los contenidos actuales de la base de datos se denomina *instancia de la base de datos.* Como sinónimos se usan los términos *extensión de la base de datos* y *estado de la base de datos.*

Los planes consisten en una enumeración de los tipos de las entidades con las que trata la base de datos y las relaciones entre esos tipos de entidades, y las formas por las cuales las entidades y relaciones a un nivel de abstracción se expresan a un nivel más bajo (más concreto). El término *esquema* es usado para referirse a los planes, por lo que hablamos de *esquema conceptual* refiriéndonos al plan para la base de datos conceptual, y *esquema físico* al plan para la base de

Este material es de uso exclusivo para los cursos impartidos por Universidad de la Empresa y asociados

datos física. El plan para una vista se denomina simplemente *subesquema.* Como sinónimo de esquema se usa también la palabra *intensión.*

1.2.3 Independencia de datos

La cadena de abstracciones, de la vista a la base de datos conceptual y desde ésta a la base de datos física, proveen dos niveles de "independencia de datos." En un sistema de bases de datos bien diseñado el esquema físico puede ser cambiado por el administrador de la base de datos sin alterar el esquema conceptual o requerir una redefinición de los subesquemas. Esta independencia se denomina *independencia física de los datos.* Debe observarse que las modificaciones a la organización de la base de datos física pueden afectar la eficiencia de los programas de aplicación, pero no requerirá la reescritura de los mismos. La ventaja de la independencia física de datos es que permite "afinar" la eficiencia de la base de datos física, permitiendo a los programas de aplicación correr como si no hubiese ocurrido cambio alguno.

La asociación entre las vistas y la base de datos conceptual provee un tipo de independencia llamada *independencia lógica de datos.* A medida que la base de datos es usada, puede volverse necesario modificar el esquema conceptual, por ejemplo, agregando información sobre tipos diferentes de entidades o información extra sobre entidades existentes. Por ejemplo, la aerolínea puede descubrir que debe proporcionar datos sobre la polución y niveles de ruido de sus vuelos, cuando la información relevante sobre sus aviones no está en la base de datos y debe agregarse.

Muchas modificaciones al esquema conceptual pueden hacerse sin afectar los subesquemas existentes, y otras modificaciones pueden hacerse si redefinimos la asociación del subesquema con el esquema conceptual. Otra vez, no son necesarios los cambios a los programas. El único tipo de cambio en el esquema conceptual que puede no verse reflejado en términos de sí mismo, es el borrado de la definición que corresponde a un subesquema. Tales cambios requieren naturalmente la reescritura o el dejar de lado de algunos programas de aplicación.

1.2.4 Lenguajes de bases de datos

En los lenguajes ordinarios de programación, las declaraciones y las sentencias ejecutables son todas partes de *un* lenguaje. En el mundo de las bases de datos, sin embargo, es normal el separar las dos funciones de declaración y computación en dos lenguajes diferentes. La motivación es que, mientras en un programa ordinario las variables existen solamente mientras el programa esta corriendo, en un sistema de bases de datos, los datos existen "para siempre", y deben ser declarados una vez para todos. Por esto, una facilidad de definición separada es conveniente.

Como hemos mencionado, el esquema conceptual es especificado en un lenguaje, suministrado como parte del DBMS, llamado *lenguaje de definición de datos.* Este lenguaje no es procedural, más bien es una notación para describir los tipos de entidades, y las relaciones entre los tipos de entidades, en términos de un modelo de datos en particular. El lenguaje de definición de datos se usa cuando la base de datos está siendo diseñada, y es usado cada vez que se modifica el diseño. No es usado para obtener o modificar a los datos. El lenguaje de definición de datos casi invariablemente tiene sentencias que describen, en ciertos términos abstractos, cuál deberá ser el formato físico de la base de datos. El diseño detallado de la base de datos física se hace mediante rutinas del DBMS que son el resultado de "compilar" las sentencias del lenguaje de definición de datos.

Este material es de uso exclusivo para los cursos impartidos por Universidad de la Empresa y asociados

La descripción de subesquemas y sus correspondencias con el esquema conceptual requieren un *lenguaje de definición de datos de subesquema,* que a menudo es bastante similar al lenguaje de definición de datos, aunque en algunos casos el lenguaje de subesquema usa un modelo de datos diferente al usado por el lenguaje de definición de datos. De hecho pueden haber varios lenguajes de esquema diferentes, cada uno con un modelo de datos distinto.

La manipulación de la base de datos requiere un lenguaje especializado, llamado *lenguaje de manipulación de datos* o *lenguaje de consulta* que permite expresar comandos como:

- *Obtener de la base de datos la cantidad de asientos disponibles en el vuelo 999 del 20 de julio,*

- *Asignar 27 a la cantidad de asientos disponibles del vuelo 123 del 31 de agosto,*

- *Encontrar algún vuelo desde el aeropuerto de Montevideo al aeropuerto de Santiago el 24 de agosto,*

- *Obtener todos los vuelos de Montevideo a Santiago del 24 de agosto.*

No todos los lenguajes de manipulación pueden obtener cantidades no especificadas de datos en un solo paso, como ocurre en la última consulta. En varios lenguajes, debemos ejecutar un comando como el penúltimo, para obtener el primer vuelo y entonces expresar *"obtener el siguiente vuelo de Montevideo a Santiago el 24 de agosto"* repetidamente hasta recibir una respuesta *"no hay mas."*

Usualmente es necesario para un programa de aplicación hacer más que manipular la base de datos; debe ejecutar una variedad de tareas computacionales ordinarias. Por ejemplo, en la sección previa mencionamos un programa de aplicación llamado *Reservas* que hacía reservaciones. Este programa debe imprimir y leer desde Terminal, tomar decisiones (¿es nula la cantidad de asientos?) y hacer aritmética (decrementar la cantidad de asientos). Por esas razones, un programa de aplicación normalmente está escrito en un lenguaje de programación convencional, como *C* o *COBOL,* llamado *lenguaje anfitrión (host language).* Los comandos del lenguaje de manipulación de datos son invocados por el programa de una de las siguientes maneras, dependiendo de las características del DBMS:

1. Los comandos del lenguaje de manipulación de datos se invocan mediante llamadas desde el lenguaje anfitrión a procedimientos suministrados por el DBMS. Los procedimientos llamados sirven como procesadores de la consulta e invocan a los niveles mas bajos del DBMS (base de datos y manejadores de archivos).

2. Los comandos son sentencias en un lenguaje que es una extensión al lenguaje anfitrión. Posiblemente haya un preprocesador que maneje las sentencias de manipulación de datos, o un compilador que maneje tanto a las sentencias del lenguaje anfitrión como las de manipulación de datos. Los comandos del lenguaje de manipulación de datos son convertidos a llamadas a procedimientos suministrados por el DBMS, por lo que la diferencia entre los enfoques (1) y (2) no es muy grande.

Internal Functions
Function: **ConectarSQL**
Description: *Conectarse a la base de datos del Hotel Caribe*
Returns
Parameters
Static Variables

Este material es de uso exclusivo para los cursos impartidos por Universidad de la Empresa y asociados

```
        Local variables
        Actions
                Set SqlDatabase = 'HOTEL'
                Set SqlUser = 'SYSADM'
                Set SqlPassword = 'SYSADM'
                If SqlConnect( hSqlHotel )
                        Call SqlSetIsolationLevel ( hSqlHotel, 'CS' )
                        Call SqlSetResultSet ( hSqlHotel, FALSE )
                        Set bConnected = TRUE
                Else
                        Call Mensaje (      'ERROR en la Conexión a la Base de
                                Datos ' )
                        Set bConnected = FALSE

Pushbutton: pbUbicar
        Message Actions
                On SAM_Click
                        Set sWhere = ''
                        Set sSelectBase =
                                'SELECT name, room, checkin, checkout, picture
                                INTO :colsNombre, :colsHabitacion,
                :coldIngreso, :coldPartida, :colsArchivo
                                FROM reservas '
                        If Not SalIsNull ( dfsNombre )
                        Set sWhere = 'WHERE name LIKE ' || '\' || dfsNombre || '\'
                        If rbONombre
                                Set sOrder = 'ORDER BY name'
                        If rbOHabitacion
                                Set sOrder = 'ORDER BY room'
                                Set sSelectHuespedes =    sSelectBase ||
                                                          sWhere ||
                                                          sOrder
                        Call       SalTblPopulate( tblHuespedes, hSqlHotel ,
                                sSelectHuespedes , TBL_FillNormal)
```

Bibliografía.

[Korth 1988]. Korth, H.F., Silberschatz A. 1988. *Fundamentos de Bases de Datos,* McGraw-Hill.

[Ullman 1982]. Ullman, J.D. 1982. *Principles of Database Systems,* Mass. Press.

[Volkmer 1987]. Volkmer, J.M. 1987. *Especificacoes Formais e Sistemas de Bancos de Dados,* I EBAI.

Este material es de uso exclusivo para los cursos impartidos por Universidad de la Empresa y asociados