Unidad Didáctica 1:

INTRODUCCIÓN A LAS BASES DE DATOS

1. Introducción

En este capítulo se presentan los sistemas de bases de datos, haciendo antes un repaso por sus predecesores, los sistemas de ficheros. Aunque los sistemas de ficheros se han quedado obsoletos, hay dos buenas razones para estudiarlos. En primer lugar, el conocer los problemas de este tipo de sistemas nos previene de volver a cometerlos. En segundo lugar, si en algún momento fuera necesario convertir un sistema de ficheros en un sistema de bases de datos, comprender cómo trabaja este sistema puede ser una ayuda esencial.

2. Sistemas de ficheros

Un ***sistema de ficheros*** es un conjunto de programas que prestan servicio a los usuarios finales. Cada programa define y maneja sus propios datos.

Los sistemas de ficheros surgieron al tratar de informatizar el manejo de los archivadores manuales con objeto de proporcionar un acceso más eficiente a los datos.

En lugar de establecer un sistema centralizado en donde almacenar todos los datos de la organización o empresa, se escogió un modelo descentralizado en el que cada sección o departamento almacena y gestiona sus propios datos.

Para comprender esto vamos a utilizar como ejemplo una empresa inmobiliaria.

> En esta inmobiliaria, el ***departamento de ventas*** se encarga de alquilar inmuebles.
>
> Por ejemplo, cuando un propietario pasa por el departamento de ventas para ofrecer en alquiler su piso, se rellena un formulario en donde se recogen los datos del piso, como la dirección y el número de habitaciones, y los datos del propietario.
>
> El departamento de ventas también se encarga de atender a los clientes que desean alquilar un inmueble. Cuando un cliente (posible inquilino) pasa por este departamento se rellena un formulario con sus datos y sus preferencias: si quiere un piso o una casa, el importe mensual que está dispuesto a pagar por el alquiler, etc. Para gestionar toda esta información, el departamento de ventas posee un sistema de información. El sistema tiene tres ficheros: fichero de inmuebles, fichero de propietarios y fichero de inquilinos.

Inum	Calle	Area	Población	Tipo	Hab	Alquiler	Pnum
IA14	Kobeta128	Centro	Bilbao	Casa	4	1600	P46
IL94	Mirivilla,24	Ronda Sur	Bilbao	Piso	4	850	P87
IG4	Sorell, 5	Norte	Bilbao	Piso	3	300	P40
IG36	Zarautz,1		Gasteiz	Piso	3	425	P93
IG21	San Blas,10		Laudio	Casa	3	350	P87
IG16	Capea,19		Bilbao	Piso	4	600	P93

Figura 1. Fichero INMUEBLES

Pnum	Nombre	Apellido	Dirección	Pref	Teléfono
P46	Begoña	Aguirre	Asensi 24,Bilbao	94	2306801
P87	Manuel	Olabarria	Av.Libertad 15,Laudio	94	6720760
P40	Alberto	Estrada	Av.del Puerto 52,Bilbao	94	2007406
P93	Yolanda	Robles	Purísima 4,Gasteiz	945	710430

Figura 2. Fichero PROPIETARIOS

Qnum	Nombre	Apellido	Dirección	Pref	Teléfono	Tipo	Alquiler
Q76	Juan	Aguirre	Barceló 47, Bilbao	94	282 5401	Piso	375
Q56	Ana	Galera	San Rafael 45, Almazora	964	551 110	Piso	300
Q74	Elena	Abasolo	Navarra 76, Bilbao	94	205 5601	Casa	700
Q62	Alicia	Atxa	Alloza 45, Bilbao	94	229 5810	Piso	550

Figura 3. Fichero INQUILINOS

El ***departamento de contratos*** se ocupa de gestionar los contratos de alquiler de los inmuebles.

Cuando un cliente desea formalizar un contrato, un empleado de la empresa rellena un formulario con los datos del inquilino y los datos del inmueble. Este formulario se pasa al departamento de contratos, que asigna un número al contrato y completa la información sobre el pago y el período del contrato. Para gestionar esta información, el departamento de contratos posee un sistema de información con tres ficheros: el fichero de los contratos, el fichero de los inmuebles alquilados y el fichero de los inquilinos que tienen en vigor un contrato de alquiler.

Cnum	Inum	Qnum	Importe	Pago	Depósito	Pagado?	Inicio	Fin	Meses
10024	IA14	Q62	600	Visa	1200	S	01/06/13	31/05/13	12
10075	IL94	Q76	350	Efectivo	700	N	01/01/13	30/06/13	6
10012	G21	Q74	550	Cheque	1100	S	01/07/12	30/06/13	12

Figura 4. Fichero CONTRATOS

Inum	Calle	Area	Población	Alquiler
IA14	Kobeta, 128	Centro	Bilbao	1600
IL94	Mirivilla, 24	Ronda Sur	Bilbao	850
IG21	San Francisco, 10		Laudio	550

Figura 5. Fichero INMUEBLES

Qnum	Nombre	Apellido	Dirección	Población	Teléfono
Q76	Juan	Aguirre	Barceló, 47	Bilbao	94 282 5401
Q74	Elena	Abasolo	Navarra, 76	Bilbao	94 205 5601
Q62	Alicia	Atxa	Alloza, 45	Bilbao	94 229 5810

Figura 6. Fichero INQUILINOS

> Cada departamento accede a sus propios ficheros mediante una serie de programas de aplicación escritos especialmente para ellos. Estos programas son totalmente independientes entre un departamento y otro, y se utilizan para introducir datos, mantener los ficheros y generar los informes que cada departamento necesita.
>
> Es importante destacar que la estructura física de los ficheros de datos y de sus registros está definida dentro de los programas de aplicación.
>
> La situación es muy similar en el resto de departamentos.
>
> En el ***departamento de nóminas*** tienen un fichero con los datos de los salarios de los empleados. Los registros de este fichero tienen los siguientes campos: número de empleado, nombre, apellido, dirección, fecha de nacimiento, salario, DNI y número de la oficina en la que trabaja.
>
> El ***departamento de personal*** tiene un fichero con los datos de los empleados. Sus registros tienen los siguientes campos: número de empleado, nombre, apellidos, dirección, teléfono, puesto, fecha de nacimiento, salario, DNI y número de la oficina en la que trabaja.

Se puede ver claramente que hay una gran cantidad de datos repetidos en los ficheros de estos departamentos, algo que siempre ocurre en los sistemas de ficheros. A raíz de esto, los sistemas de ficheros presentan una serie de **inconvenientes**:

a) ***Separación y aislamiento de los datos***. Cuando los datos se separan en distintos ficheros, es más complicado acceder a ellos, ya que el programador de aplicaciones debe sincronizar el procesamiento de los distintos ficheros implicados para asegurar que se extraen los datos correctos.

b) ***Duplicación de datos.*** La redundancia de datos existente en los sistemas de ficheros hace que se desperdicie espacio de almacenamiento y lo que es más importante: puede llevar a que se pierda la consistencia de los datos. Se produce una inconsistencia cuando copias de los mismos datos no coinciden.

c) ***Dependencia de datos***. Ya que la estructura física de los datos (la definición de los ficheros y de los registros) se encuentra codificada en los programas de aplicación, cualquier cambio en dicha estructura es difícil de realizar. El programador debe identificar todos los programas afectados por este cambio, modificarlos y volverlos a probar, lo que cuesta mucho tiempo y está sujeto a que se produzcan errores. A este problema, tan característico de los sistemas de ficheros, se le denomina también *falta de independencia de datos lógica-física*.

d) ***Formatos de ficheros incompatibles.*** Ya que la estructura de los ficheros se define en los programas de aplicación, es completamente dependiente del lenguaje de programación. La incompatibilidad entre ficheros generados por distintos lenguajes hace que los ficheros sean difíciles de procesar de modo conjunto.

e) ***Consultas fijas y proliferación de programas de aplicación***. Desde el punto de vista de los usuarios finales, los sistemas de ficheros fueron un gran avance comparados a los sistemas manuales. A consecuencia de esto, creció la necesidad de realizar distintos tipos de consultas de datos. Sin embargo, los sistemas de ficheros son muy dependientes del programador de aplicaciones: cualquier consulta o informe que se quiera realizar debe ser programado por él. En algunas organizaciones se conformaron con fijar el tipo de consultas e informes, siendo imposible realizar otro tipo de consultas que no se hubieran tenido en cuenta a la hora de escribir los programas de aplicación. En otras organizaciones hubo una proliferación de programas de aplicación para resolver todo tipo de consultas, hasta el punto de desbordar al departamento de proceso de datos, que no daba abasto para validar, mantener y documentar dichos programas.

3. Sistemas de bases de datos

Los inconvenientes de los sistemas de ficheros se pueden atribuir a dos factores:

- La definición de los datos se encuentra codificada dentro de los programas de aplicación, en lugar de estar almacenada aparte y de forma independiente.
- No hay control sobre el acceso y la manipulación de los datos más allá de lo impuesto por los programas de aplicación.

Para trabajar de un modo más efectivo, surgieron las *bases de datos.*

Una ***base de datos*** es un conjunto de datos almacenados entre los que existen relaciones lógicas y ha sido diseñada para satisfacer los requerimientos de información de una empresa u organización. En una base de datos, además de los datos, también se almacena su descripción.

La base de datos es un gran almacén de datos que se define una sola vez y que se utiliza al mismo tiempo por muchos departamentos y usuarios. En lugar de trabajar con ficheros desconectados e información redundante, todos los datos se integran con una mínima cantidad de duplicidad. La base de datos no pertenece a un departamento, se comparte por toda la organización. Además, la base de datos no sólo contiene los datos de la organización, también almacena una descripción de dichos datos. Esta descripción es lo que se denomina *metadatos*, se almacena en el *diccionario de datos* o catálogo y es lo que permite que exista independencia de datos lógica-física.

El tamaño y la complejidad de la bases de datos depende del problema que se esté resolviendo, por lo que podemos tener una base de datos personal para guardar las direcciones y teléfonos de nuestros mejores amigos, o bien tener una base de datos que almacene todos los datos clínicos de los usuarios de la Seguridad Social. Por tanto, tal cantidad de información, tiene que organizarse y controlarse para que la información pueda ser accedida y manipulada cuando sea necesario. Además, su almacenamiento ha de ser eficiente, así como las operaciones de manipulación que se llevan a cabo sobre los datos.

Resumiendo lo que acabamos de contemplar, podemos definir la base de datos como :

"Colección o depósito de datos integrados, almacenados en soporte secundario (no volátil) y con redundancia controlada. Los datos, que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de ellos, y su definición (estructura de la base de datos) única y almacenada junto con los datos, se ha de apoyar en un modelo de datos, el cual ha de permitir captar las interrelaciones y restricciones existentes en el mundo real. Los procedimientos de actualización y recuperación, comunes y bien determinados, facilitarán la seguridad del conjunto de los datos."

4. Distintos niveles de abstracción de una base de datos

Un objetivo importante de los SGBD es proporcionar a los usuarios una **visión abstracta de los datos** , es decir, el sistema debe ocultar los detalles sobre cómo se acceden y se manipulan los datos.

Esto da origen a la creación de **niveles de abstracción** , que muestran distintas visiones de la complejidad de la representación de la información, es decir, ocultan los detalles de almacenamiento. En 1975, el comité ANSI-SPARC (American National Standard Institute - Standards Planning and Requirements Committee) propuso una arquitectura que permite ver una base de datos a tres niveles de abstracción denominados nivel físico, nivel conceptual y nivel externo.

- **Nivel físico o interno.** Es el nivel de abstracción más bajo, y describe cómo se almacenan realmente los datos.
- **Nivel conceptual o lógico** . Es el nivel que describe qué información se almacena en la base de datos, y cómo está relacionada dicha información. La definición de una estructura de datos a nivel conceptual puede suponer la creación de varias estructuras a nivel físico (creación de archivos indexados por varios campos, por ejemplo).
- **Nivel externo o de vistas** . Es el nivel de abstracción más alto, y en el que sólo se describen partes de la base de datos, ya que no todos los usuarios pueden acceder a la misma parte de la base de datos. Para facilitar la interacción del usuario con el

sistema, se definen varios niveles de visión, de forma que cada uno represente lo que cada usuario o grupo de usuarios necesita.

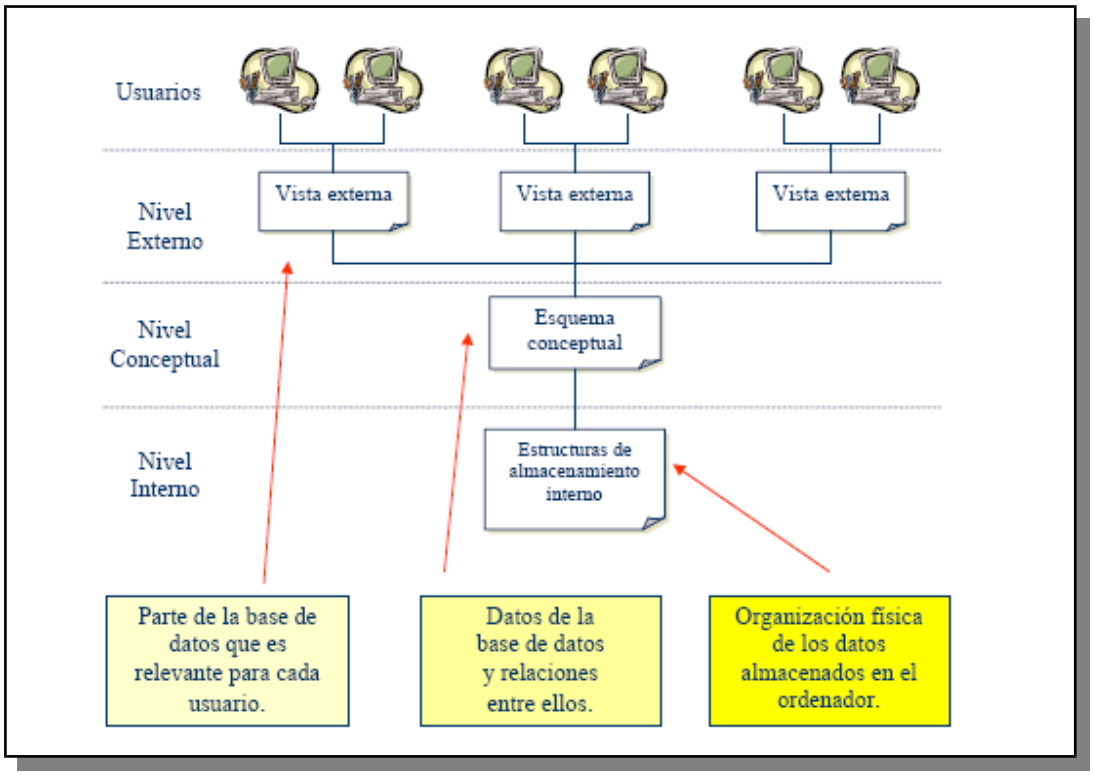

Figura 7. Niveles de abstracción de una BD

A partir de estos niveles de abstracción es posible establecer una analogía entre dichos niveles de abstracción y los tipos de datos de los lenguajes de programación, como se muestra a continuación.

La mayoría de los lenguajes de programación de alto nivel permiten una declaración de tipos semejante a esta (hecha en este caso en C):

```
typedef struct DatosAlumno
{
  char nombre[50];
  char apellido1[50];
  char apellido2[50];
  char dni[8];
  char direccion[50];
  char localidad[50];
  char codigoPostal[50];
};

typedef struct NotasAlumno
{
```

```
    char dni[8];
    char notas[5][10];
};
```

En el nivel físico, estos registros pueden describirse como un bloque de posiciones de memoria. En el nivel conceptual estos registros quedan descritos mediante la declaración anterior. Finalmente, en el nivel externo se definen varias **vistas** de la base de datos, ya que no todos los usuarios tienen acceso a la información sobre las notas de los alumnos.

En general, al usuario no le interesa saber dónde está cada campo dentro del registro (nivel conceptual), ni en qué posición de memoria está guardada la información que está consultando (nivel físico), sino que lo que interesa, es que cuando quiera introducir las calificaciones de los alumnos se realice en los campos y registros adecuados, y que cuando haga referencia a la nota de SGBD de 2º del alumno X se devuelva la información correcta.

Con esta arquitectura a tres niveles conseguimos la independencia de datos. Hay dos niveles de independencia de datos:

- **Independencia física de datos** . Es la capacidad de modificar el esquema físico sin necesidad de modificar los programas de aplicación. Esto es, por ejemplo, modificar el tamaño de un campo o modificar los índices del archivo.
- **Independencia lógica de datos** . Es la capacidad de modificar el esquema conceptual sin necesidad de modificar los programas de aplicación. Si, por ejemplo, se reduce la base de datos eliminando un tipo de datos, los esquemas externos que no se refieran a esta parte de la base de datos no deberán verse afectados. La independencia lógica es más difícil de conseguir que la independencia física, ya que los programas de aplicación suelen ser muy dependientes de la estructura lógica de los datos a los que acceden (se han creado a partir de ella).

5. Sistemas Gestores de Bases de Datos

En las secciones previas se han descrito las características que debe tener una base de datos y que las diferencia de los sistemas de almacenamiento convencionales.

Es importante conocer la diferencia entre lo que es una base de datos y lo que es un *Sistema de Gestión de Bases de Datos,* términos que se confunden muy a menudo cuando se está trabajando con la información haciendo uso de esta tecnología.

Hasta estos momentos se ha estado tratando únicamente el término de bases de datos. Cuando se habla de bases de datos se habla de información que está almacenada cumpliendo toda una serie de características y restricciones como las que se han expuesto en este capítulo.

Pero para que la información pueda ser almacenada como se ha descrito y el acceso a la misma satisfaga las características exigidas a una base de datos para ser denominada como tal, es necesario que exista una serie de procedimientos -un sistema software- que sea capaz de llevar a cabo tal labor. A este sistema software es a lo que se le denomina *Sistema de Gestión de Bases de Datos.* Así, un **SGBD (Sistema de Gestión de Bases de**

Datos) es un conjunto de programas (en realidad, un sistema software) de propósito general que facilita el proceso de definición, construcción y manipulación de bases de datos para usos diversos.

6. Funciones de los SGBD

Cuando una empresa u organización decide adquirir una licencia de un SGBD para su infraestructura informática antes ha debido pasar por una fase de análisis de sus necesidades y de revisión de las posibilidades que cada producto brinda. En una situación ideal, los SGBD deberían ofrecer las mismas funcionalidades, pero como todo en la vida, esto no sucede así. En esta sección se describen las principales características reseñables de un SGBD.

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

- Un SGBD debe proporcionar a los usuarios la capacidad de **almacenar** datos en la base de datos, **acceder** a ellos y **actualizarlos** . Esta es la función fundamental de un SGBD.

- Un SGBD debe proporcionar un **catálogo** en el que se almacenen las descripciones de los datos y que sea accesible por los usuarios. Este catálogo es lo que se denomina diccionario de datos y contiene información que describe los datos de la base de datos (metadatos). Normalmente, un diccionario de datos almacena:

    - Nombre, tipo y tamaño de los datos.
    - Nombre de las relaciones entre los datos.
    - Restricciones de integridad sobre los datos.
    - Nombre de los usuarios autorizados a acceder a la base de datos.
    - Esquemas externos, conceptual e interno, y correspondencia entre los esquemas.
    - Estadísticas de utilización.

- Un SGBD debe proporcionar un mecanismo que garantice que todas las actualizaciones correspondientes a una determinada **transacción** se realicen, o que no se realice ninguna. Una transacción es un conjunto de acciones que cambian el contenido de la base de datos. Una transacción en el sistema informático de la empresa inmobiliaria sería dar de alta a un empleado o eliminar un inmueble. Una transacción un poco más complicada sería eliminar un empleado y reasignar sus inmuebles a otro empleado. En este caso hay que realizar varios cambios sobre la base de datos. Si la transacción falla durante su realización, por ejemplo porque falla el hardware, la base de datos quedará en un estado inconsistente. Algunos de los cambios se habrán hecho y otros no, por lo tanto, los cambios realizados deberán ser deshechos para devolver la base de datos a un estado consistente.

- Un SGBD debe proporcionar un mecanismo que asegure que la base de datos se actualice correctamente cuando varios usuarios la están actualizando concurrentemente. Uno de los principales objetivos de los SGBD es el permitir que varios usuarios tengan **acceso concurrente** a los datos que comparten. El acceso concurrente es relativamente fácil de gestionar si todos los usuarios se dedican a leer

datos, ya que no pueden interferir unos con otros. Sin embargo, cuando dos o más usuarios están accediendo a la base de datos y al menos uno de ellos está actualizando datos, pueden interferir de modo que se produzcan inconsistencias en la base de datos. El SGBD se debe encargar de que estas interferencias no se produzcan en el acceso simultáneo.

- Un SGBD debe proporcionar un mecanismo capaz de **recuperar la base de datos** en caso de que ocurra algún suceso que la dañe.

- Un SGBD debe proporcionar un mecanismo que garantice que sólo los **usuarios autorizados** pueden acceder a la base de datos. La protección debe ser contra accesos no autorizados, tanto intencionados como accidentales.

- Un **SGBD debe ser capaz de integrarse** con algún software de comunicación. Muchos usuarios acceden a la base de datos desde terminales. En ocasiones estos terminales se encuentran conectados directamente a la máquina sobre la que funciona el SGBD. En otras ocasiones los terminales están en lugares remotos, por lo que la comunicación con la máquina que alberga al SGBD se debe hacer a través de una red. En cualquiera de los dos casos, el SGBD recibe peticiones en forma de mensajes y responde de modo similar. Todas estas transmisiones de mensajes las maneja el gestor de comunicaciones de datos. Aunque este gestor no forma parte del SGBD, es necesario que el SGBD se pueda integrar con él para que el sistema sea comercialmente viable.

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

- Un SGBD debe proporcionar una serie de herramientas que permitan **administrar la base de datos** de modo efectivo. Algunas herramientas trabajan a nivel externo, por lo que habrán sido producidas por el administrador de la base de datos. Las herramientas que trabajan a nivel interno deben ser proporcionadas por el distribuidor del SGBD. Algunas de ellas son:

    - Herramientas para importar y exportar datos.
    - Herramientas para monitorizar el uso y el funcionamiento de la base de datos.
    - Programas de análisis estadístico para examinar las prestaciones o las estadísticas de utilización.
    - Herramientas para reorganización de índices.

- Herramientas para aprovechar el espacio dejado en el almacenamiento físico por los registros borrados y que consoliden el espacio liberado para reutilizarlo cuando sea necesario.

7. Usuarios de la base de datos

Hay cuatro grupos de personas que intervienen en el entorno de una base de datos: el administrador de la base de datos, los diseñadores de la base de datos, los programadores de aplicaciones y los usuarios.

- El ***administrador de la base de datos*** (DBA: DataBase Administrator) se encarga del diseño físico de la base de datos y de su implementación, realiza el control de la seguridad y de la concurrencia, mantiene el sistema para que siempre se encuentre operativo y se encarga de que los usuarios y las aplicaciones obtengan buenas prestaciones. El administrador debe conocer muy bien el SGBD que se esté utilizando, así como el equipo informático sobre el que esté funcionando.

- Los ***diseñadores de la base de datos*** realizan el diseño lógico de la base de datos, debiendo identificar los datos, las relaciones entre datos y las restricciones sobre los datos y sus relaciones. El diseñador de la base de datos debe tener un profundo conocimiento de los datos de la empresa y también debe conocer sus *reglas de negocio*. Las reglas de negocio describen las características principales de los datos tal y como las ve la empresa.

  Para obtener un buen resultado, el diseñador de la base de datos debe implicar en el desarrollo del modelo de datos a todos los usuarios de la base de datos, tan pronto como sea posible. El diseño lógico de la base de datos es independiente del SGBD concreto que se vaya a utilizar, es independiente de los programas de aplicación, de los lenguajes de programación y de cualquier otra consideración física.

- Una vez se ha diseñado e implementado la base de datos, los ***programadores de aplicaciones*** se encargan de implementar los programas de aplicación que servirán a los usuarios finales. Estos programas de aplicación son los que permiten consultar datos, insertarlos, actualizarlos y eliminarlos. Estos programas se escriben mediante lenguajes de tercera generación o de cuarta generación.

- Los ***usuarios finales*** son los clientes de la base de datos: la base de datos ha sido diseñada e implementada, y está siendo mantenida, para satisfacer sus requisitos en la gestión de su información.