

# Módulo 1: Introducción a las bases de datos.

# La explosión de la información

## *La explosión de la información*

- se denomina así al enorme **crecimiento actual** de las **necesidades** de la información y a la mayor **disponibilidad** de este recurso

puede conducir, si no se ponen los medios para evitarlo, a una *polución informativa*

- ☹ fenómeno análogo al de la contaminación del aire, en el que la información, al **perder sus cualidades**, no puede cumplir sus objetivos
- ☹ llegando incluso a ser **más nociva que beneficiosa** para sus destinatarios

# Cualidades de la información

✎ Para evitar el peligro de la polución informativa se debe exigir a la información un conjunto de **cualidades**

☹ que mantengan su **valor comunicativo** y son básicamente:

& *precisión,*

& *oportunidad,*

& *compleción,*

& *significado e*

& *integridad*

☹ Todas ellas en el grado que exija cada sistema concreto.

# Precisión

✎ Se puede definir como el porcentaje de **información correcta** sobre la información total del sistema

☹ Sin embargo, el tratamiento por ordenador no puede mejorar la calidad de los datos elaborados

& lo único que puede hacer la máquina es señalar ciertos errores o **incompatibilidades**

- como máximo, sustituir el dato detectado como erróneo, por otro que no tenga error aparente, es decir, que sea **coherente**

↩ & Por tanto, si queremos que los resultados del ordenador sean precisos debemos también suministrarle datos precisos

- no pudiendo pretender en los resultados una precisión superior a la que tenían los datos de entrada

☹ Una precisión baja lleva inevitablemente a una falta de credibilidad del usuario hacia la información que se le proporciona

# Oportunidad

↪ La oportunidad se refiere al tiempo transcurrido

- ☹ desde el momento en que se produjo el hecho que originó el dato
- ☹ hasta el momento en el que la información se pone a disposición del usuario

& Al igual que ocurre con la precisión, también la oportunidad depende de cada aplicación

- \* Por ejemplo, para un censo en el que se manejan datos de carácter bastante estable, un tiempo de proceso de meses no le resta oportunidad a la información
- \* en cambio esta demora en la obtención de los indicadores de coyuntura, como el IPC, sería inadmisibile
- En general, el valor de la información disminuye con el tiempo
  - \* la pérdida de valor será más o menos rápida dependiendo del tipo de información.
- En investigaciones históricas, por el contrario, la información gana con el transcurso del tiempo

# Compleción

✍ La información ha de ser completa para poder cumplir sus fines

- Por ejemplo, un informe que se emite al objeto de que un directivo tome una decisión ha de contener todos los elementos informativos necesarios para apoyar dicha decisión

☹ Lo que se suele pretender en los sistemas de información es alcanzar un nivel que se considere suficiente, el cual dependerá de dos factores:

& de los datos existentes en el sistema de información y

& de los que el sistema es capaz de localizar ante una consulta concreta

- En este segundo factor influirán
  - ✳ la flexibilidad e idoneidad del lenguaje de recuperación y
  - ✳ el acierto en la formulación de la consulta
- Así pues, la completión no es sólo función de la información en sí misma, sino también de otros factores, tanto técnicos como humanos.

# Significado y coherencia

✎ La información que se suministra al usuario debe ser también *significativa*

☹ es decir, ha de poseer el máximo contenido semántico posible

☹ ya que sin él no sería verdadera información

& Esto lleva a que ha de ser comprensible e interesante, lo que supone no proporcionar a los usuarios grandes masas de información que por su volumen no puedan ser asimiladas

✎ Toda la información contenida en el sistema debe ser

☹ **coherente** en sí misma y

☹ **consistente** con las *reglas semánticas* propias del mundo real al que ha de representar lo más fielmente posible

& esta cualidad, que en las bases de datos se conoce con el nombre de *integridad*, coincide **sólo en parte** con el concepto que hemos definido como precisión.

# Seguridad

✍ La información ha de ser **protegida**

& frente a su **deterioro** por causas

- físicas o
- lógicas

& y frente a **accesos no autorizados**

☹ La seguridad de la información esta adquiriendo una gran relevancia

& muy especialmente con la difusión de las nuevas posibilidades de las comunicaciones

☹ Actualmente el concepto de seguridad comprende

& *confidencialidad*,

& *disponibilidad* e

& *integridad*



# Cualidades de la información

- ☹ Al implantar un sistema de información es preciso tener muy en cuenta todos estos requisitos de la información
  - & buscando el punto de equilibrio para alcanzar los objetivos del sistema a un coste aceptable
    - ya que cuantas más cualidades reúna la información
    - más se incrementará su coste de obtención y tratamiento
- ☹ Además, unas cualidades pueden resultar incompatibles con otras
  - Por ejemplo, pretender una gran precisión lleva consigo generalmente una pérdida de oportunidad

# Concepto de Sistema de Información

- ✎ Un *sistema* es un conjunto de elementos relacionados ordenadamente entre sí de acuerdo a ciertas reglas, y que contribuyen a determinado objetivo
- ✎ Un *sistema de información (SI)* es un sistema cuyo objetivo es aportar a la organización a la que sirve la información necesaria para el cumplimiento de sus fines
- ✎ Toda organización necesita para su funcionamiento un conjunto de informaciones que han de transmitirse
  - ☹ entre sus distintos elementos y
  - ☹ generalmente también, desde y hacia el exterior del sistema

# SI informales y formales

- ☹ Una parte de esta comunicación se realiza por medio de contactos **interpersonales** entre los empleados constituyendo el sistema de información *informal*
- ☹ Pero este tipo de flujo de información, cuando se trata de organismos complejos, se muestra insuficiente y costoso
  - & siendo preciso disponer de un sistema de información *formal*
    - \* también llamado *organizacional*
  - que aporte al organismo la información necesaria de forma *eficaz* y *eficiente*
- ☹ De ahora en adelante, y siempre que no se indique expresamente lo contrario, al mencionar el término SI nos estaremos refiriendo a SI formales

# Eficacia y eficiencia

- ✎ Constituyen dos parámetros fundamentales para evaluar el comportamiento de un sistema.
  - ☹ Llamamos **eficacia** al grado en que se cumplen los objetivos del sistema
    - & Es, en cierto modo, una medida referida a las relaciones **externas** del sistema
  - ☹ La **eficiencia** está más enfocada hacia operaciones y relaciones **internas** del sistema
    - & mide el grado de optimización en el uso de los recursos disponibles
  - ☹ Aunque son parámetros distintos, están muy interrelacionados
    - & ¿hasta qué punto la optimización de los recursos (eficiencia) no aumenta la eficacia?

# SI informatizados

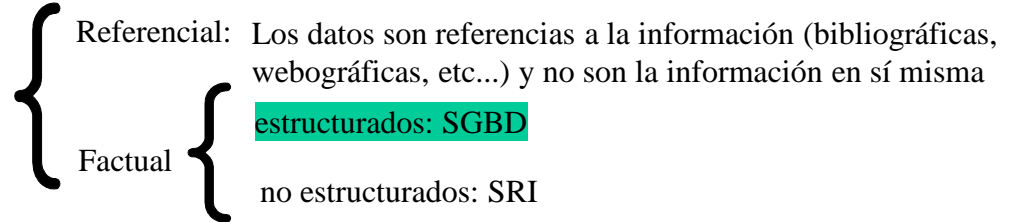
- ✎ Aun cuando los SI podrían no estar informatizados
  - ☹ siendo tratados manualmente
- ✎ los SI actuales se apoyan en técnicas informáticas
  - & los tratamientos y
  - & la recuperación de la información
  - ☹ se realizan, **a menudo**, por medio de *sistemas de gestión de bases de datos*
- ✎ A partir de ahora, el término SI designará a un SI *formal e informatizado*
  - ☹ salvo que se indique expresamente lo contrario

# Componentes básicos de un SI

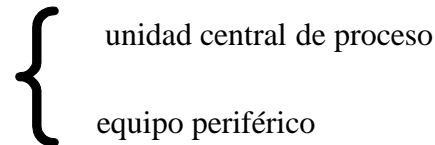
S  
I  
S  
T  
E  
M  
A  
  
D  
E  
  
I  
N  
F  
O  
R  
M  
A  
C  
I  
Ó  
N



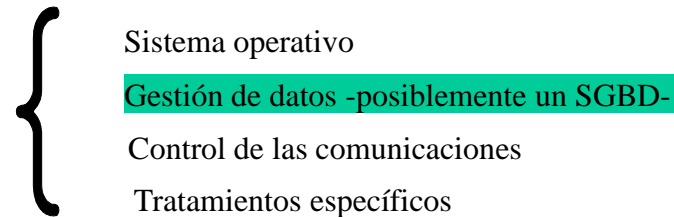
Contenido  
-datos y su descripción-



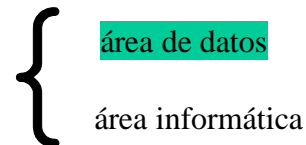
Equipo físico  
-hardware-



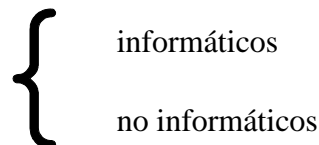
Soporte lógico  
-software-



Administrador



Usuarios



# Estática y dinámica

✎ En un SI se pueden distinguir **dos** perspectivas diferentes pero evidentemente **interrelacionadas**

☹ La perspectiva *estática*

& que está relacionada con la **información** de la organización

☹ y la perspectiva *dinámica*

& que está relacionada con las operaciones que la organización realiza para el **tratamiento** de dicha información


# Fases del desarrollo de un SI: Análisis

## Fase 1: *Análisis*

 **Investigación:** Descubrir el conjunto de requerimientos

- de **información** y
- de **procesos**

 que necesita la organización para cumplir sus fines

 **Diseño conceptual:** Obtener una **representación de la realidad** que capture sus propiedades

- **estáticas** y
- **dinámicas**

 Esta imagen de la realidad se denomina *esquema conceptual*



# Fases del desarrollo de un SI:

## Diseño

### Fase 2: *Diseño*

☹ **Diseño lógico:** Una vez conocidas las **técnicas de gestión de datos** que se van a utilizar:

- sistemas de **ficheros**,
- sistemas de gestión de **bases de datos**,
- etc...

& se **traduce** el esquema **conceptual** en términos de dicha tecnología, dando lugar a

- ✳ un *esquema lógico* (expresión de la estática)
- ✳ un conjunto de *transacciones* (expresión de la dinámica)

☹ **Diseño físico:**

- teniendo en cuenta los detalles de la **representación física** de los datos
- y atendiendo a **criterios de eficacia y eficiencia**

& se obtiene el *esquema físico*

- ✳ como un refinamiento del esquema **lógico**

# Fases del desarrollo de un SI: Implantación

## Fase 3: Implantación

- ☹ Esta fase supone la incorporación del SI diseñado a la organización
  - & con la puesta en marcha de la base de datos
    - y consecuentemente la **carga de datos**
  - & y el desarrollo de aplicaciones
    - incluyendo la puesta en marcha de los **programas** de manipulación

# Fases del desarrollo de un SI

## 1ª fase: *Análisis*

*Investigación:*

Requerimientos  
de **información**

Requerimientos  
de **procesos**

**Modelado semántico**

*Diseño conceptual:*

Esquema **conceptual**

*Estática*

*Dinámica*

## 2ª fase: *Diseño*

*Diseño lógico:*

Esquema **lógico**

Esquema de  
**transacciones**

**Tecnología de gestión de datos**

*Diseño físico:*

Esquema **físico**

**Sistema de gestión de BD**

## 3ª fase: *Implantación*

Carga de la base  
de **datos**

**Programas**

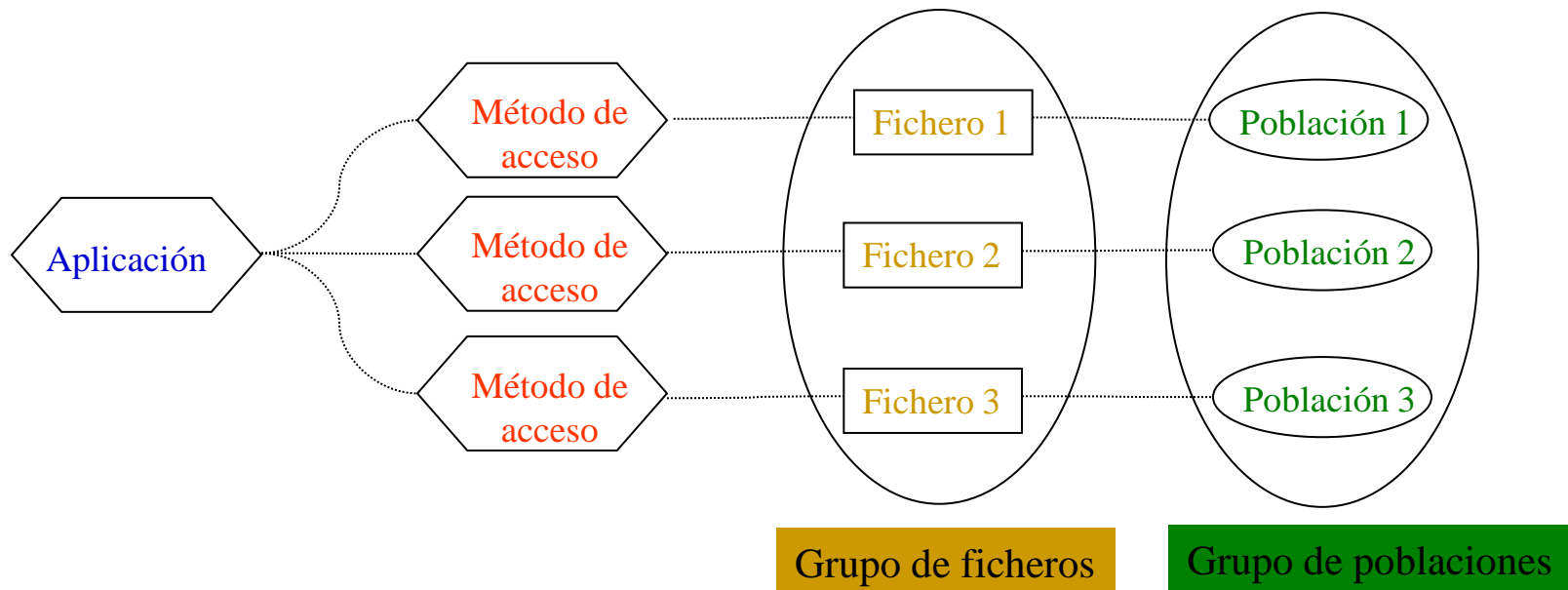
En cada fase se recurre a **herramientas y técnicas descriptivas** que permiten **representar** totalmente el **sistema** en sus **dos perspectivas**

# La tecnología de gestión de datos

- Una de las primeras decisiones a tomar en el **diseño lógico** es la elección de la *tecnología de gestión de datos* a usar
- En principio, podría optarse por
  - & un sistema de gestión de **ficheros** (SGF),
  - & un sistema de gestión de **bases de datos** (SGBD) o incluso
  - & un sistema de **recuperación de información** (SRI)
- ☹ Aquí se descartará, por ahora, el uso de SRI
  - & dado que sus características **escapan** del ámbito de los sistemas de información *factual estructurada*
- La elección estaría entonces entre los SGF clásicos y los SGBD
  - ☹ Excepto en casos concretos, hoy en día la elección parece clara
    - decantándose a **favor** del uso de **bases de datos**

# Sistemas de gestión de ficheros

- La mayoría de las *aplicaciones* requieren varios ficheros
- Cada fichero representa a una *población*
- La aplicación se comunica con los ficheros a través de los *métodos de acceso*



# Sistemas de gestión de ficheros

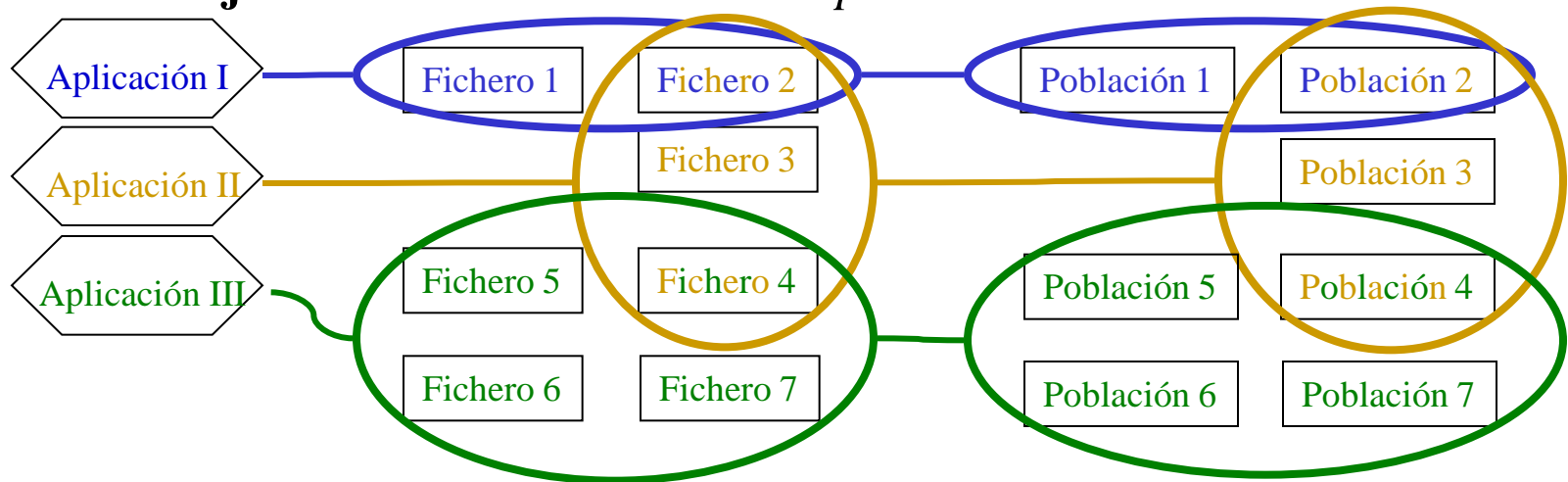
En un SI suelen coexistir **varias aplicaciones**

- Cada aplicación usa un grupo de ficheros
  - \* y consecuentemente se relaciona con un grupo de poblaciones

☹ Una **población** puede interesar a **más de una aplicación**

- y en tal caso existirán **solapamientos** entre los grupos de ficheros

& sin embargo, esas *aplicaciones* pueden **requerir diferentes conjuntos de datos** de la misma *población*



Familia de aplicaciones

Familia de ficheros

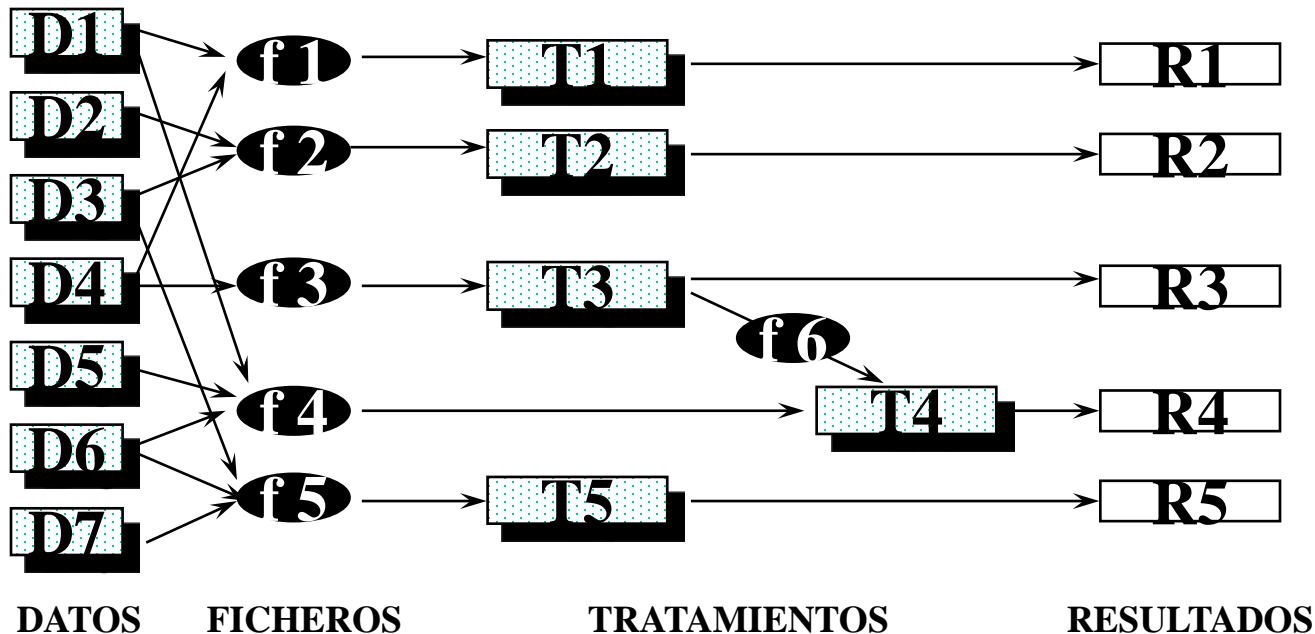
Familia de poblaciones

# Sistemas de gestión de ficheros orientados al proceso

☹ Se hace **énfasis** en los **tratamientos** que reciben los **datos**

& Los datos se almacenan en ficheros diseñados para una determinada aplicación (*existe interdependencia entre la estructura del fichero y la aplicación*)

& Los **datos no se comparten** sino que se *duplican* cuando las correspondientes aplicaciones los necesitan (existe *redundancia*)

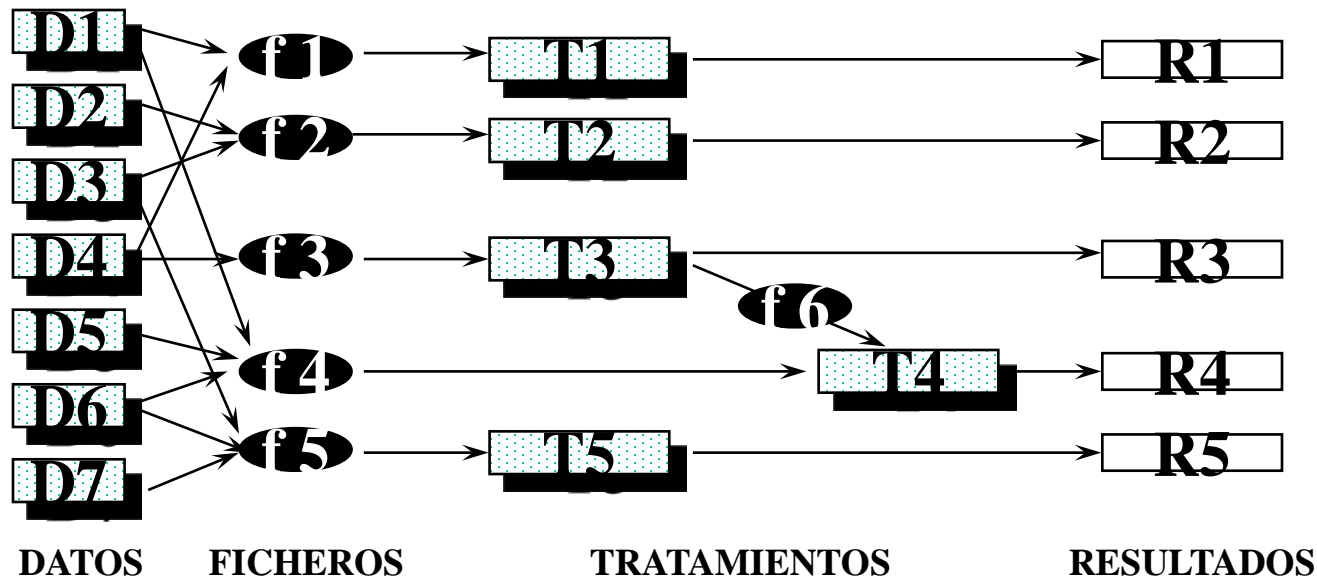




Grupo de Estructuras de Datos

# Sistemas de gestión de ficheros orientados al proceso: Ejemplo

- ☹ Sean dos secciones diferentes de la ULPGC: la sección de personal y la secretaría de la EUI
- ☹ Sus correspondientes **aplicaciones** tendrán **visiones diferentes** de la **población** de profesores, que se traducirán en **dos ficheros** diferentes:
  - &Personal: Profesor = {nombre, dirección, categoría, n° cuenta, estado civil, n° de hijos}
  - &Secretaría: Profesor = {nombre, dirección, categoría, departamento, horario, asignaturas}
- Existe redundancia, al duplicarse tres de los datos de la misma población
- Los datos incluidos en cada fichero dependen de la aplicación





# Sistemas de gestión de ficheros orientados al proceso

## ✎ Inconvenientes causados por la redundancia

& Desperdicio de **memoria** secundaria

& Posibilidad de inconsistencias causadas por **actualizaciones incoherentes** de la información duplicada

& **Aumento** de los **tiempos** de **proceso** para poder mantener **coherentemente** los datos **repetidos**

☹ Ello no implica que siempre sea posible o incluso deseable eliminar toda la redundancia

& En ocasiones existen motivos técnicos o de la propia organización para mantener almacenadas varias copias de los mismos datos

☹ Sin embargo, si que es conveniente minimizar, documentar y controlar con mucho cuidado esa posible redundancia

# Sistemas de gestión de ficheros orientados al proceso

✎ La interdependencia entre fichero y aplicación produce una **ventaja**:

☹ Cada aplicación **obtiene exclusivamente los *datos* requeridos** sin tener que acceder al conjunto completo de datos de la población

& Téngase en cuenta que una **aplicación** puede **necesitar *solamente*** un **pequeño conjunto de datos** de una **población** que posea una **gran cantidad de *descriptores***

- produciendo **retardos *innecesarios*** en los accesos a memoria secundaria

# Sistemas de gestión de ficheros orientados a poblaciones

➤ Una posible modificación consistiría en mantener un único fichero por población

& sin que cada aplicación tenga su propia copia particularizada

☹ Ahora *la estructura del fichero depende de la población*

☹ Los **datos** se *comparten* y **no** se **duplican** cuando las correspondientes aplicaciones los necesitan (ya **no** existe *redundancia*)

# Sistemas de gestión de ficheros orientados a poblaciones: Ejemplo

☹ En el ejemplo presentado anteriormente, las aplicaciones correspondientes a las dos secciones de la ULPGC en cuestión siguen teniendo **visiones diferentes** de la **población** de profesores:

&Personal: Profesor = {nombre, dirección, categoría, nº cuenta, estado civil, nº de hijos}

&Secretaría: Profesor = {nombre, dirección, categoría, departamento, horario, asignaturas}

☹ Sin embargo estas visiones **no** estarán **representadas** por **sendos ficheros**, sino que existirá **un único fichero** para la *población* de profesores:

&Profesor = {nombre, dirección, categoría, nº cuenta, estado civil, nº de hijos, departamento, horario, asignaturas}

- Ya no existe redundancia
- Existe un único fichero con los datos que la población requiera

# Sistemas de gestión de ficheros orientados a poblaciones

✎ Al **no existir redundancia desaparecen** los **problemas** que esta aportaba al sistema, a saber:

- & **Desperdicio de memoria** secundaria

- & Posibilidad de **inconsistencias** causadas por **actualizaciones incoherentes** de la información duplicada

- & **Aumento** de los **tiempos de proceso** para poder mantener **coherentemente** los datos **repetidos**

☹ Aunque conviene aquí aclarar que existen otros tipos de redundancias más sutiles

- & que no vienen dadas por la elección de la tecnología de gestión de datos

- & y que pueden producir problemas similares a los presentados

# Sistemas de gestión de ficheros orientados a poblaciones

➤ Ahora la **estructura** del **fichero depende** de la **población**

☹ Al leer el *fichero unificado*, **cada aplicación** interesada en la población **accederá** al **conjunto completo de descriptores** de la población

&posibilitando **retardos innecesarios** en el **acceso a memoria secundaria**

# Sistemas de gestión de ficheros orientados a poblaciones

☹ Además, los **cambios en el diseño del fichero unificado** **afectarán** a **todas** las **aplicaciones** interesadas en esa población

- **estén o no interesadas por dicho cambio**

- Por ejemplo, al añadir el *nº de seguridad social* a la población de *profesores* deberá modificarse la aplicación de la sección de *personal*



- \* **justificadamente**, puesto que está **interesada** en ese dato y la aplicación de *secretaría* de la EUI

- \* **pese** a que es una información que **no utiliza**

& En los sistemas de gestión de ficheros orientados al proceso **sólo afectaría** a las **aplicaciones interesadas** en el **cambio** de diseño

- que serían las que necesitarían modificar sus respectivos ficheros

- En el ejemplo, sólo se modificaría la aplicación de la sección de personal,

- \* puesto que el **fichero** usado por la **aplicación** de la **secretaría** de la EUI **no se retocaría**

☹ Ello **provoca**

& **falta** de **flexibilidad** del SI para enfrentar **variaciones** en los **requerimientos** de los usuarios

& **alto coste** del **mantenimiento** del software

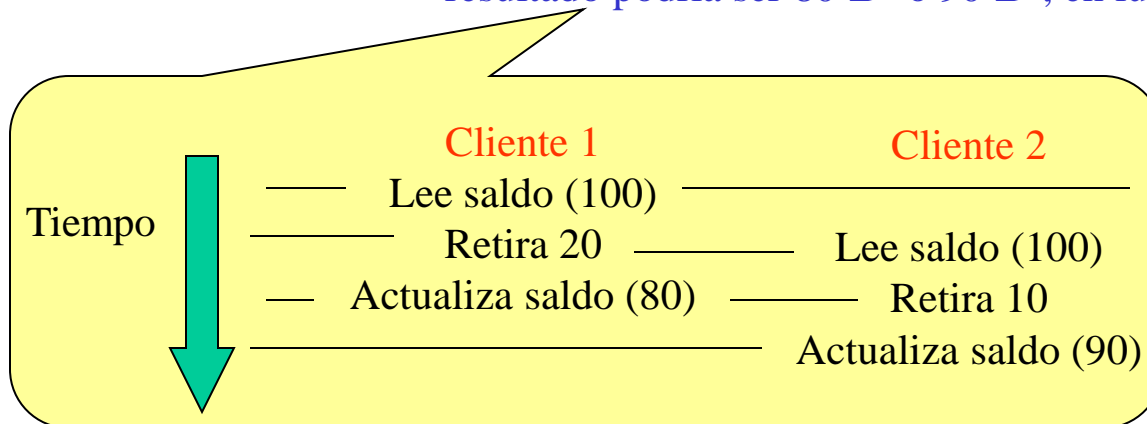
# Inconvenientes adicionales de los sistemas de gestión de ficheros

En general, la elección de un *sistema de gestión de ficheros* como **tecnología** de gestión de datos **dificulta**, en mayor o menor medida, las siguientes **funciones**:

& el control de los **accesos concurrentes** (*integridad operacional*)

- la interacción de actualizaciones concurrentes puede producir datos inconsistentes

\* Sea una cuenta bancaria con 100 ₧ de saldo. Si dos clientes retiran fondos (por ejemplo, 10 ₧ y 20 ₧ respectivamente) casi al mismo tiempo, el resultado podría ser 80 ₧ ó 90 ₧, en lugar del resultado correcto de 70 ₧





# Inconvenientes adicionales de los sistemas de gestión de ficheros

## & la recuperación de ficheros (*disponibilidad*)

- Un sistema informático está sujeto a **fallos** provocados por errores físicos (roturas de disco, fallos de alimentación, ...) ó lógicos (errores de software, ...)
- Si estos fallos provocan **actualizaciones incompletas** o **erróneas** hay que devolver a los ficheros a un **estado anterior correcto** y esto puede ser muy laborioso

## & la vigilancia de los accesos (*confidencialidad*)

## & el control de la integridad de la información (*integridad semántica*)

☹ **todas ellas relacionadas con el concepto genérico de seguridad**

⌚ **resta capacidad de reacción frente a demandas inesperadas de información**

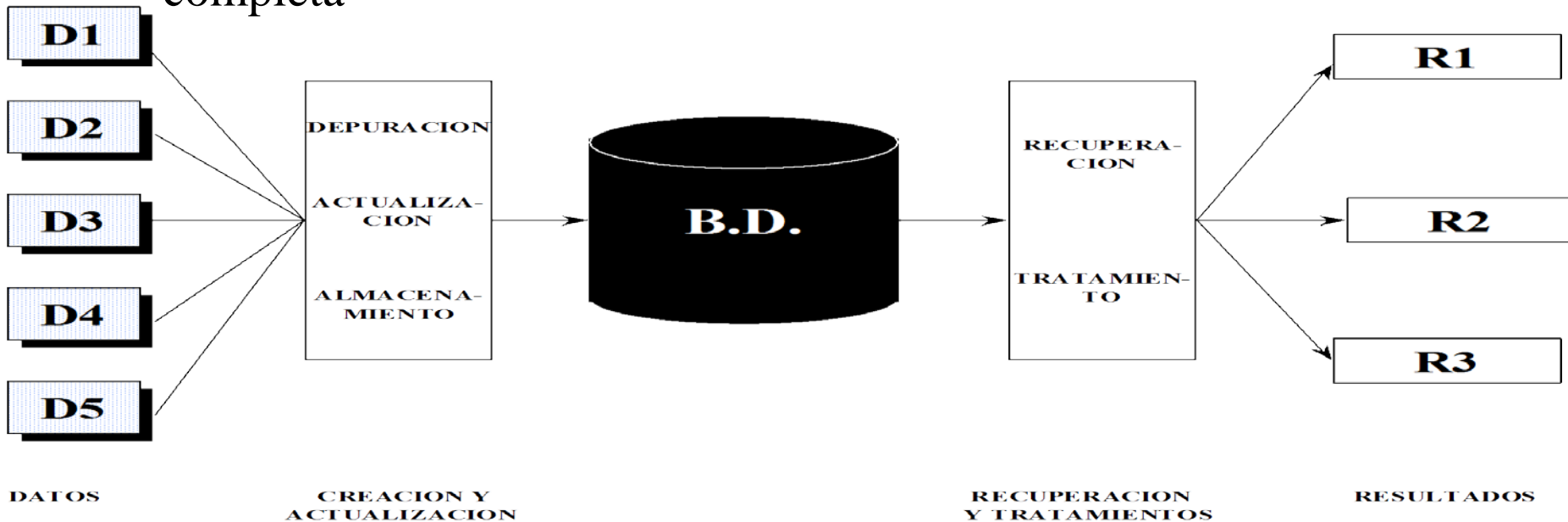
⌚ **y hace inalcanzable la implantación de verdaderos sistemas de información orientados a la toma de decisiones**

# Sistemas de gestión de bases de datos

De lo anterior se deduce la **necesidad** de un nuevo enfoque en el que los datos se **almacenan** y **mantienen** en un conjunto **estructurado** (*base de datos*)

☹ **sin redundancia**

☹ **diseñado** para **toda** la **organización** y no para una aplicación completa





Grupo de Estructuras de Datos

# Sistemas de gestión de bases de datos

✎ Mantiene internamente un fichero por población

☹ como no hay redundancia no aparecen los problemas asociados

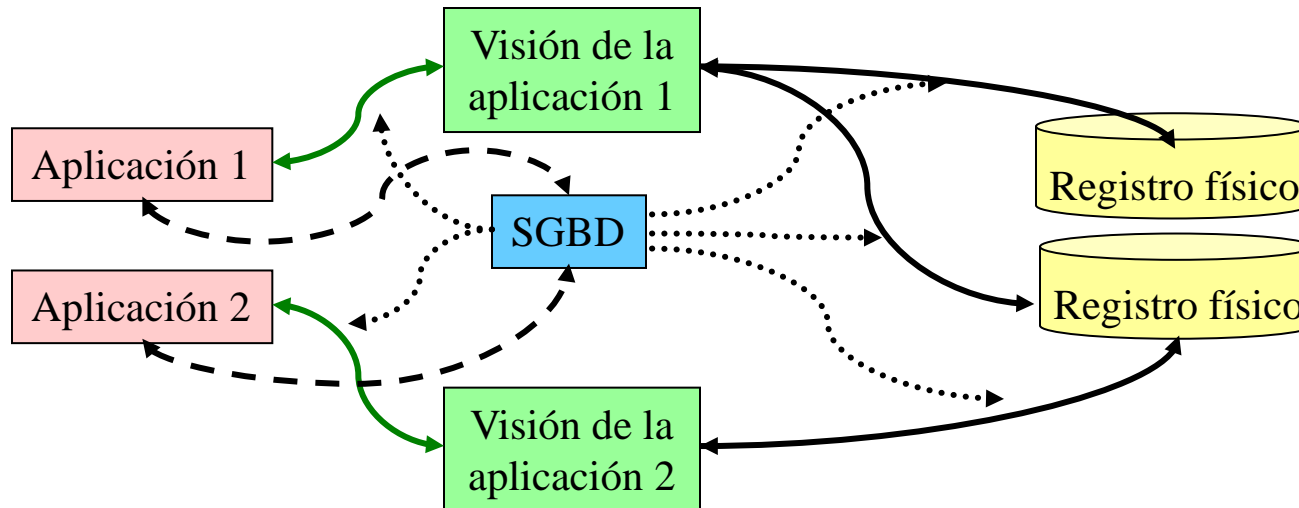
☹ **suministra a cada aplicación sólo los datos necesarios**

& Toma el *registro físico interno*

- lo **descompone**

- y **entrega** a la **aplicación** un nuevo *registro lógico* de acuerdo a las **necesidades de la aplicación**

✳ Incluso puede proporcionar a una aplicación datos de poblaciones diferentes



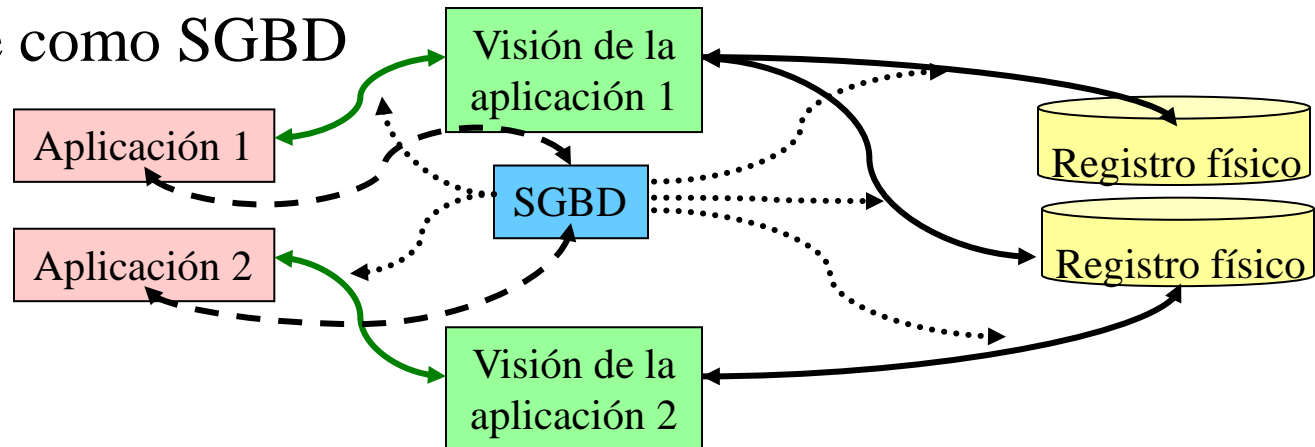
# Sistemas de gestión de bases de datos

⤴ Ante los **cambios de diseño** del fichero que representa a una población

☹ **sólo** habrá que **modificar** las **aplicaciones** que se **interesen** por dichos **cambios**

& serán aquellas que **modifiquen** su *visión* para recoger las alteraciones

⤴ La herramienta software desarrollada para lograr estos objetivos se conoce como SGBD



# Concepto de *base de datos*

- ✍ “Colección de datos interrelacionados almacenados en conjunto sin redundancias perjudiciales o innecesarias; su finalidad es servir a una aplicación o más, de la mejor manera posible; los datos se almacenan de modo que resulten independientes de los programas que los usan; se emplean métodos bien determinados para incluir nuevos datos y para modificar o extraer los datos almacenados”. (Martin, 1975).
- ✍ “Colección o depósito de datos, donde los datos están lógicamente relacionados entre sí, tienen una definición y descripción comunes y están estructurados de una forma particular. Una base de datos es también un modelo del mundo real y, como tal, debe poder servir para toda una gama de usos y aplicaciones”. (Conference des Statisticiens Européens, 1977).
- ✍ “Conjunto de datos de la empresa memorizado en un ordenador, que es utilizado por numerosas personas y cuya organización está regida por un modelo de datos”. (Flory, 1982).

# Concepto de *base de datos*

- “Conjunto estructurado de datos registrados sobre soportes accesibles por ordenador para satisfacer simultáneamente a varios usuarios de forma selectiva y en tiempo oportuno”. (Delobel, 1982).
- “Colección no redundante de datos que son compartidos por diferentes sistemas de aplicación”. (Howe, 1983).
- “Colección integrada y generalizada de datos, estructurada atendiendo a las relaciones naturales de modo que suministre todos los caminos de acceso necesarios a cada unidad de datos con objeto de poder atender todas las necesidades de los diferentes usuarios”. (Deen, 1985).

# Concepto de *base de datos*

“Conjunto de ficheros maestros, organizados y administrados de una manera flexible de modo que los ficheros puedan ser fácilmente adaptados a nuevas tareas imprevisibles”. (Frank, 1988).

“Colección de datos interrelacionados”. (Elsmani y Navathe, 1989).

# Concepto de *base de datos*

*“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”*



# Concepto de *base de datos*

☞ Merece la pena realizar algunas **observaciones** sobre el contenido de esta última definición:

☹ Dada la relevancia que tienen en el mundo real las *interrelaciones* entre los datos

- es imprescindible que la base de datos sea capaz de **almacenar** estas interrelaciones

& En el mundo real existen, además, *restricciones semánticas*, a las que se está concediendo una importancia creciente

- en los sistemas actuales, tienden a **almacenarse junto** con los **datos**, al **igual** que ocurre con las **interrelaciones**

# Concepto de *base de datos*

☹ La **redundancia** de los datos debe ser *controlada*

& de forma que no existan **duplicidades** perjudiciales ni innecesarias



& y que las redundancias **físicas**

\* **convenientes** muchas veces a fin de responder a objetivos de **eficiencia**  
sean **tratadas por el mismo sistema**

\* de modo que **no** puedan producirse **inconsistencias**

- Esto podría resumirse diciendo que en las bases de datos **no** debe existir ***redundancia lógica***

aunque **sí** se **admite** cierta ***redundancia física*** por motivos de **eficiencia**

& Por tanto, un dato se actualizará *lógicamente* por el usuario de forma **única**


- y el **sistema** se preocupará de **cambiar físicamente** todos aquellos campos en los que el dato estuviese repetido en caso de existir **redundancia física**

& Es lo que se denomina también ***redundancia controlada*** por el sistema.

# Objetivos de los SGBD

## *Independencia* de los datos:

- ☹ Las **aplicaciones** que se comuniquen con el SGBD deben verse **afectados lo menos posible** por **cambios** efectuados en la **estructura de los datos** que **no utilizan**

 Esta aporta **flexibilidad** para la adaptación de los SI a la evolución de la organizaciones

## *Integridad* de los datos:

- ☹ La información almacenada en la base de datos debe **satisfacer** ciertas **restricciones** de consistencia
  - definidas explícitamente por los usuarios
  - el SGBD determina si las actualizaciones suponen o no la violación de dichas restricciones
- ☹ Se debe asegurar la **correcta ejecución** de los **accesos concurrentes**
- ☹ Ha de ofrecerse la posibilidad de **recuperación** de la base de datos **frente a posibles deterioros**

# Objetivos de los SGBD

- ✎ ***Proteger*** a los datos frente a **accesos desautorizados**
- ✎ Sencillez en la ***interfaz*** frente a los usuarios y aplicaciones
  - ☹ proporcionando, entre otras funciones, adecuados **métodos de acceso**
- ✎ Aumentar la ***disponibilidad*** y la ***trasparencia*** de la información existente
  - ☹ todos los datos que se encuentran en la base se deben relacionar en un catálogo o diccionario
    - & que puede ser ampliamente difundido y accedido por medios informáticos


# Objetivos de los SGBD


## Mayor *valor informativo*:


 La base de datos, como reflejo del mundo real recoge las *interrelaciones* entre los datos


- el valor informativo del conjunto es superior a la suma del valor informativo de los elementos individuales que lo constituyen

 actúa el efecto de sinergia

 En la base se incluye, integrada con los datos, la *semántica* de los mismos

 No todos los SGBD facilitan las mismas prestaciones a estos respectos

 pero la tendencia actual es conseguir que la descripción de los datos incluida dentro del sistema sea lo más completa posible

 y que el diccionario que la contiene, accedido por el SGBD, sea capaz de almacenar y tratar el máximo de semántica.

# Inconvenientes de los SGBD

Estos inconvenientes cobran su verdadera dimensión en el caso de grandes bases de datos  
nunca para pequeños sistemas instalados en ordenadores personales

☹ Instalación costosa:

& La implantación de un sistema de bases de datos puede llevar consigo un coste elevado

- tanto en equipo físico

- \* nuevas instalaciones o ampliaciones

- como en el lógico

- \* sistemas operativos, programas, compiladores, etc... necesarios para su uso

además del propio coste de adquisición y mantenimiento del SGBD

☹ Necesidad de personal especializado.

☹ Implantación larga y difícil

☹ Falta de rentabilidad a corto plazo

# Niveles de abstracción en los SI

✎ En los SI se puede observar la existencia de dos estructuras distintas:

- ☹ la *lógica*: transmite la **vista** que el **usuario** tiene de los datos que usa
- ☹ y la *física*: transmite la **forma** en que se encuentran esos **datos** en el **almacenamiento**

# Niveles de abstracción en los SGBD

✎ En las bases de datos aparece un nuevo nivel de abstracción que se ha denominado de diversas maneras:

&nivel **conceptual**,

&lógico **global**,

&etc ...

☹ Esta estructura **lógica** intermedia pretende una **representación global** de los datos

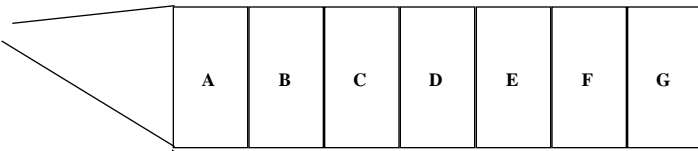
que se **interponga** entre las estructuras lógica y física de la arquitectura a dos niveles

siendo **independiente**, tanto **del SGBD** a utilizar, como **de cada usuario** en particular



# Niveles de abstracción en los SGBD

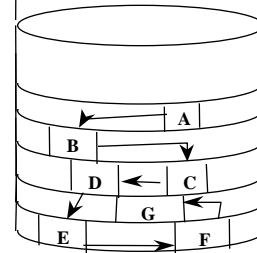
- ☹ La estructura **lógica de usuario** o **esquema externo** es la visión que tiene de la base de datos cada usuario en particular
- ☹ la estructura **lógica global**  
& también denominada **esquema conceptual** responde al enfoque del conjunto de la empresa
- ☹ la estructura **física** (**esquema interno**) es la forma cómo se organizan los datos en el almacenamiento físico



ESTRUCTURA  
LOGICA DE USUARIO  
-esquema externo-



ESTRUCTURA  
LOGICA  
GLOBAL



ESTRUCTURA  
FISICA  
-esquema interno-



Grupo de Estructuras de Datos

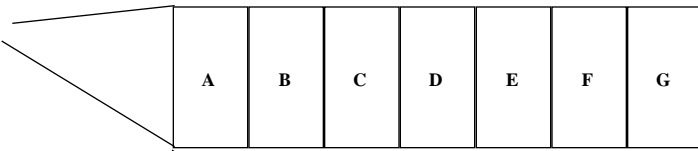
# Niveles de abstracción en los SGBD

## ☹ La terminología es confusa

&según los **autores**, grupos de **estandarización** o **modelos** un mismo **concepto** recibe diversos nombres

- \* el **esquema externo ANSI** se corresponde con la **vista** del **modelo relacional**,
- \* llamándose **subesquema** en el modelo **Codasyl**

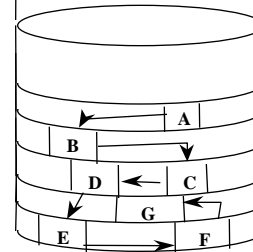
## ☹ La estructuración de una base de datos en estos **tres niveles** de *abstracción* tiene como principal **objetivo** conseguir la **independencia** entre datos y aplicaciones



ESTRUCTURA  
LOGICA DE USUARIO  
-esquema externo-



ESTRUCTURA  
LOGICA  
GLOBAL  
-esquema conceptual-



ESTRUCTURA  
FISICA  
-esquema interno-

# Niveles de abstracción en los SGBD según la arquitectura ANSI/SPARC

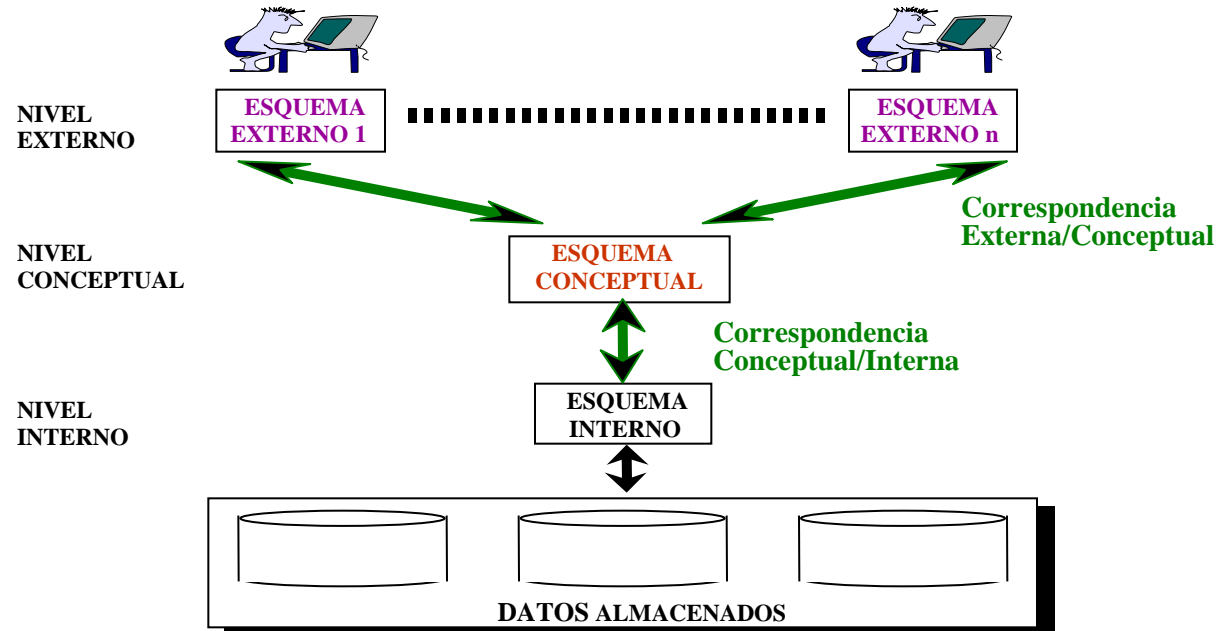
☹ Se establece el término *esquema conceptual* para la descripción **global** de los datos

& de la cual **deriva** una *colección* de *esquemas externos*

☹ Las **transformaciones** de unos esquemas en otros

& denominadas *funciones de correspondencia*

las lleva a cabo el SGBD



# Esquema *conceptual*

.vs.

# Esquema *lógico*

✎ El *esquema conceptual* ha de ser **independiente** del **SGBD** a utilizar:

- ☹ En el estado **actual** de la técnica de las bases de datos
  - ↪ no existe ningún *modelo conceptual* **general** y accesible desde **cualquier tipo de SGBD**
  - ↪ que nos permita **definir** el *esquema conceptual*
- ☹ Por este motivo es **preferible distinguir dos esquemas** en lugar del *esquema conceptual*

# Esquema *conceptual*

.vs.

# Esquema *lógico*

☹ El *esquema conceptual*:

& Visión **lógica global** del **SI**, **independiente** del **SGBD** que se utilice, e incluso de que se utilice o no una BD

☹ El *esquema lógico*:

& Visión **lógica global** de la **BD** del **SI**, **expresada** en términos del **tipo** de **SGBD** que se vaya a utilizar

✍ Ambos son esquemas globales de tipo lógico

# Tipos de usuarios informáticos de una BD

➤ **Usuarios *informáticos*:** Tienen a su cargo

& las tareas de **creación** y **mantenimiento** de la base de datos

& la realización de los procedimientos y **programas** que necesiten los usuarios finales

➤ Entre ellos se puede distinguir:

☹ **Diseñadores:** Tienen la responsabilidad de

& **identificar los datos** que han de estar contenidos en la base de datos

● de acuerdo con las **necesidades** que les manifiesten los **usuarios**

& **determinar las estructuras** más apropiadas para conseguir satisfacer estas necesidades

# Tipos de usuarios informáticos de una BD

- ☹ **Administradores:** Su misión es la **vigilancia y gestión de los datos**
- & Debe velar para que éstos **no** se **destruyan** **ni** se **contaminen**, perdiendo su **confidencialidad, disponibilidad e integridad**
  - & Será el **responsable** de establecer el sistema de **autorizaciones** de acceso y deberá **coordinar y controlar** su uso
  - & Tendrá a su cargo el **SGBD** y otras **herramientas** relacionadas con el mismo
  - & Deberá ocuparse del **buen funcionamiento** de todo el sistema
    - **sin** que se produzcan **paradas** y
    - de modo que se proporcionen los **adecuados tiempos de respuesta**
  - & En muchas organizaciones es la misma persona o grupo de personas las que tienen a su cargo las funciones de diseño y de administración
    - aunque se trata de papeles muy distintos que convendría diferenciar claramente

# Tipos de usuarios informáticos de una BD

## ☹ *Analistas y Programadores:*

& Tienen a su cargo el *análisis* y la *programación* de las **tareas** que **no** pueden ser **llevadas a cabo por** los **usuarios finales**

- han de **desarrollar** distintos procedimientos y **programas**
- que ponen a disposición de los usuarios finales a fin de facilitarles su trabajo



# Tipos de usuarios finales de una BD

✎ **Usuarios *finales*:** Tienen que acceder a los datos porque los necesitan para llevar a cabo su actividad

& A diferencia de los usuarios informáticos, su **interés** suele estar **centrado** en los **datos**

✎ Existen también distintas clases de usuarios finales:

## ☹ **Habituales:**

& Suelen hacer consultas y/o actualizaciones a la base de datos como parte habitual de su trabajo

& **Utilizan** en general **menús** previamente **preparados** por analistas y/o **programadores** (*tareas formalizables*)

✱ Aunque en algunos casos pueden usar lenguajes sencillos para el acceso a la base de datos

& Es preciso distinguir, dentro de este grupo, a los **operadores de entrada de datos**, cuya labor consiste en actualizar la base de datos

- se les prepara menús de actualización

✱ aunque con la **diferencia** respecto a otros usuarios habituales de que tienen **exigencias** muy estrictas respecto a los **tiempos de respuesta**

# Tipos de usuarios finales de una BD

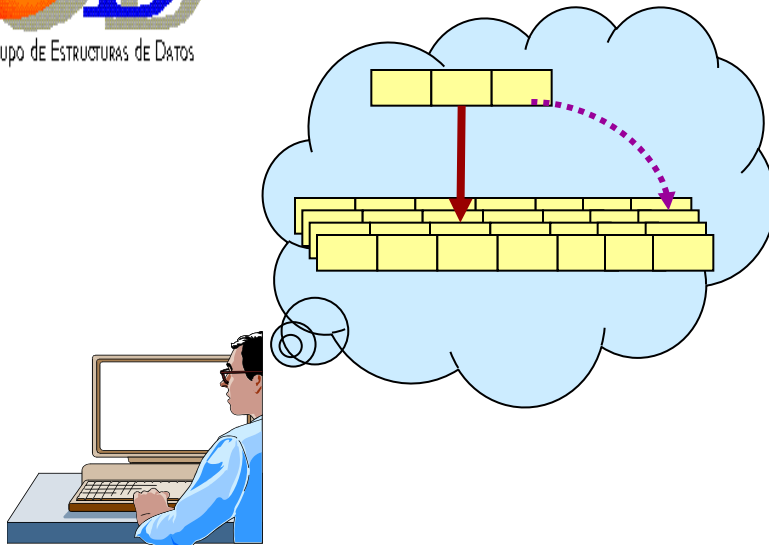
## ☹ *Esporádicos:*

& No utilizan habitualmente la base de datos

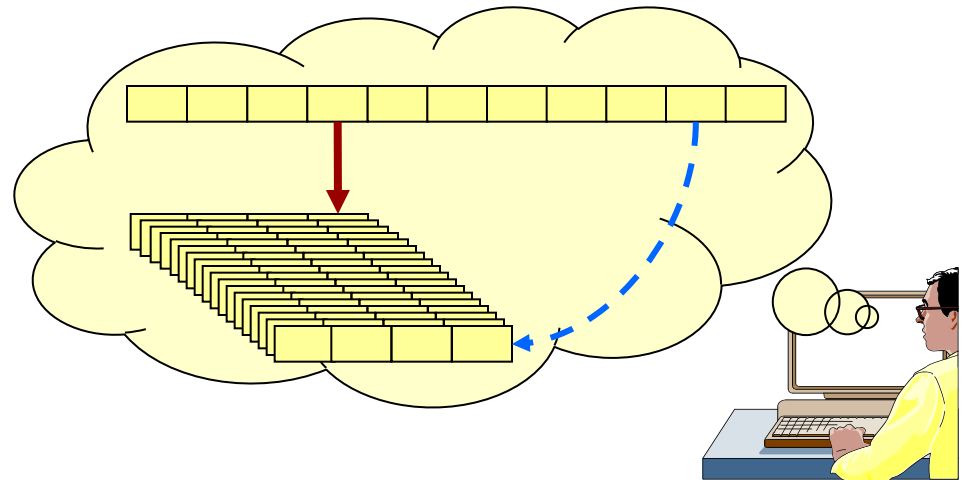
& Sus tareas no suelen ser *formalizables*

- por lo que no pueden ser atendidas por medio de **menús** previamente preparados
- y suelen **requerir** *inquiridores* de la base de datos con **lenguajes sencillos** pero **potentes**

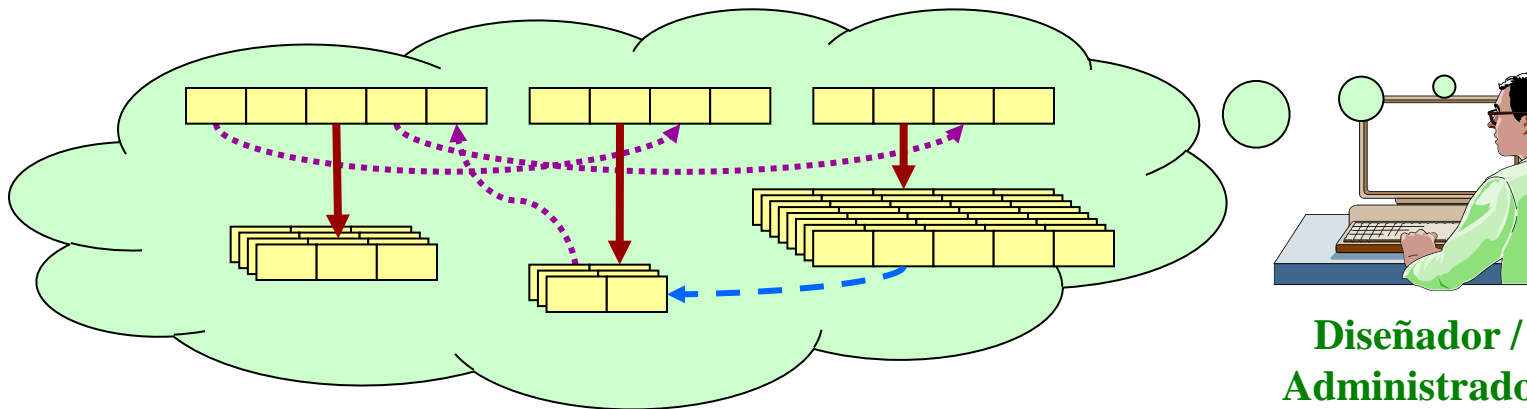
# Diferentes vistas de los datos



**Programador 1**



**Programador 2**



**Diseñador /  
Administrador**

# Operaciones típicas sobre una base de datos

## A) SOBRE EL CONJUNTO DE LA BASE

- Creación
- Reestructuración
- Consulta a la totalidad ► Se recuperan:
  - ⇒ todos los datos de la base ó
  - ⇒ todos los de un determinado tipo  
(todos los empleados para la nómina)

## B) SOBRE REGISTROS CONCRETOS

- Inserción
  - Borrado
  - Modificación
  - Consulta selectiva
- } Actualización

# Funciones básicas de un SGBD

## DESCRIPCION

### Permite describir:

- Los elementos de datos con
  - Su estructura
  - Sus interrelaciones
  - Sus validaciones

### A tres niveles:

- Externo
- Lógico Global
- Interno

*Mediante un LDD (DDL)*

## MANIPULACION

### Permite:

- Buscar
  - Añadir
  - Suprimir
  - Modificar
- datos de la base

*Mediante un LMD (DML)*

### Lo cual supone:

- Definir un criterio de selección (responsabilidad usuario)
- Definir el esquema externo a recuperar (responsabilidad usuario)
- Acceder a la estructura física (responsabilidad del sistema)

## CONTROL

- En ocasiones se le denomina *utilización*
- A veces no aparece al considerarse integrada en las dos anteriores)

- Reúne las interfaces que necesitan los usuarios para comunicarse con la base
- Suministra procedimientos para el administrador



Grupo de Estructuras de Datos

# Lenguajes y facilidades contenidas en un SGBD

## LENGUAJES DE DEFINICION DE DATOS (LDD)

- Definición de datos  $\left\{ \begin{array}{l} \text{-Externo} \\ \text{-Global} \\ \text{-Interno} \end{array} \right\}$  lógico

## LENGUAJES DE MANIPULACION DE DATOS (LMD)

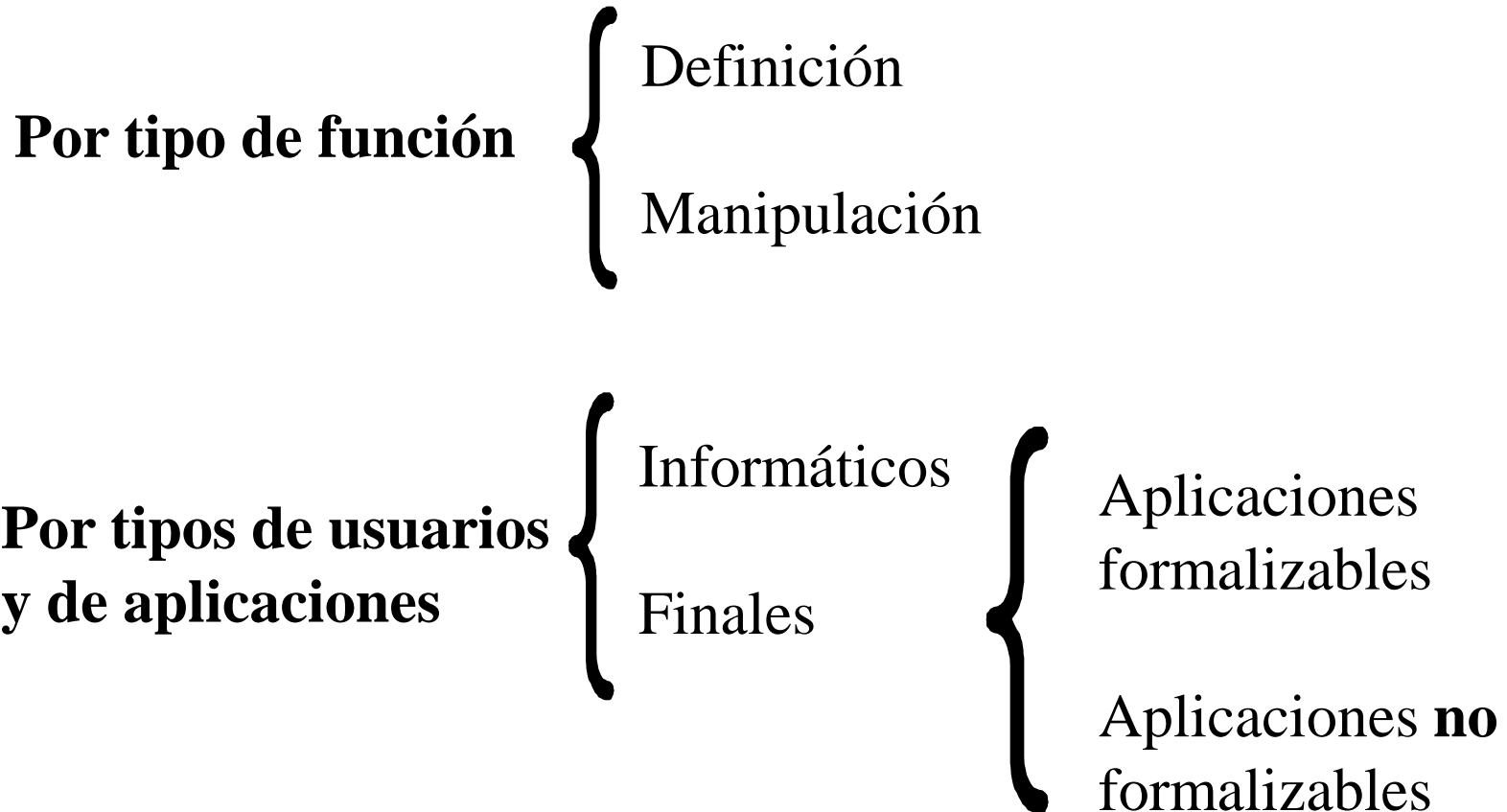
- Manipulación de datos  $\left\{ \begin{array}{l} \text{-Recuperación} \\ \text{-Actualización} \end{array} \right\}$

## PROCEDIMIENTOS PARA EL ADMINISTRADOR

- Reorganizaciones
- Copias de seguridad
- Estadísticas
- Cargas de ficheros

.....

# Tipología de los lenguajes de los SGBD



# Clasificación de los lenguajes de datos

{ Huésped  
Autocontenido

{ Muy procedimental  
Poco procedimental

{ Diferido (por lotes)  
Conversacional (interactivo)

{ Registro a registro (navegacional)  
Conjunto de registros (especificación)





# Estructura simplificada de un programa escrito en un lenguaje anfitrión que llama a un LMD huésped

.....

## SENTENCIAS DECLARATIVAS

.....

**declaración de áreas de E/S para las transferencias de datos desde/hacia la BD**  
**declaración de áreas para la comunicación de mensajes**

.....

**LLAMADA A LA VISTA DE USUARIO (interacción con la BD)**

.....

## SENTENCIAS DE PROCESO

.....

**LLAMADA A LA SENTENCIA DEL LMD (interacción con la BD)**  
**comprobación del contenido del área de mensajes**

.....

# Ejemplo de sentencia en SQL autocontenido

SELECT *nombre, apellido*

FROM **persona**

WHERE *fecha\_nac* = “28/11/65”;

# Lenguajes de datos

☹ En general, los lenguajes de tipo **huésped**:

&son **procedimentales**,

&se explotan en **diferido** y

&actúan **registro a registro**

☹ mientras que los **autocontenidos**:

&suelen ser **no procedimentales**,

&se usan en **conversacional** y

&recuperan **conjuntos de registros**

● Por *ejemplo*:

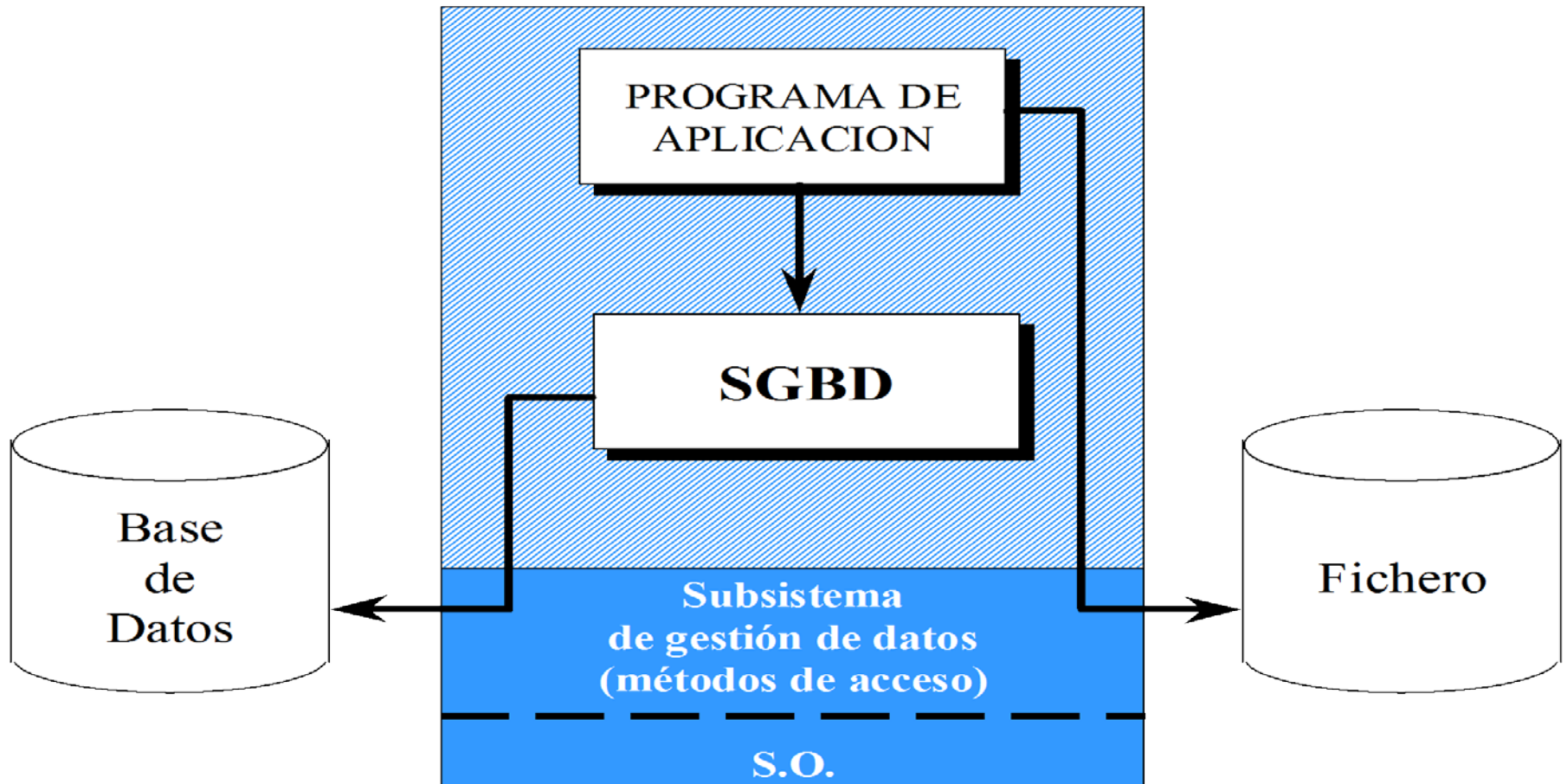
✱ el *DL/I* es un lenguaje de tipo huésped, muy procedimental, diferido y navegacional

✱ el lenguaje *SQL* al que hemos hecho referencia anteriormente, es poco procedimental, de especificación (actúa sobre conjunto de registros) y puede usarse en modo autocontenido o como lenguaje huésped desde un lenguaje anfitrión, ya que goza de la propiedad dual

# Forma de acceso a un fichero

.vs.

# Forma de acceso a una base de datos

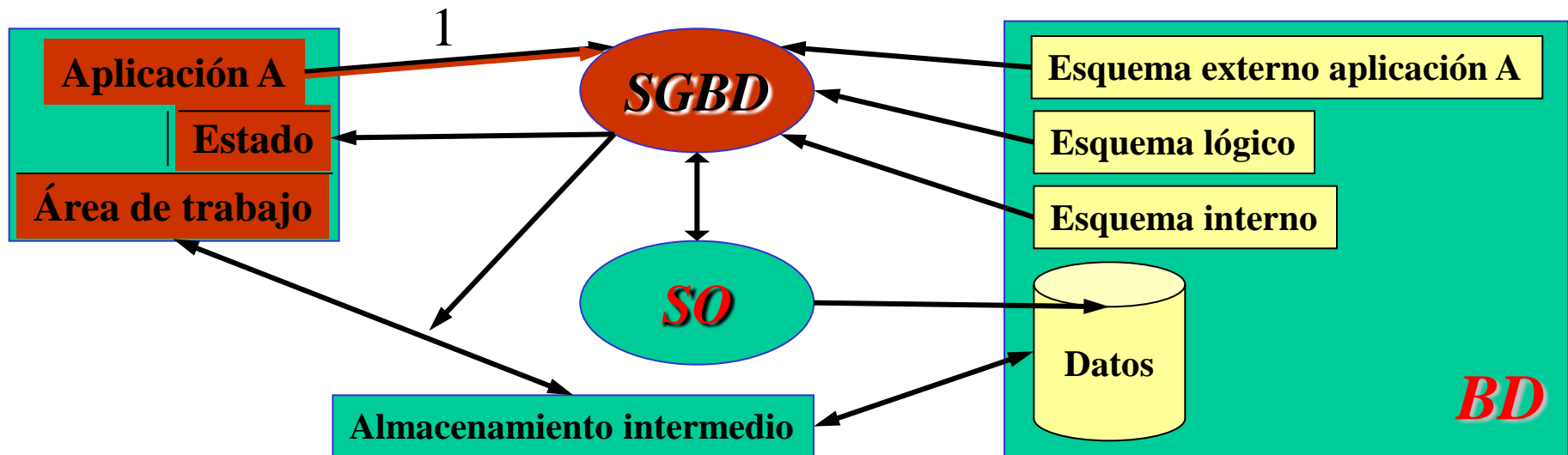


# Interacción del SGBD con el SO

➤ Cada aplicación tiene:

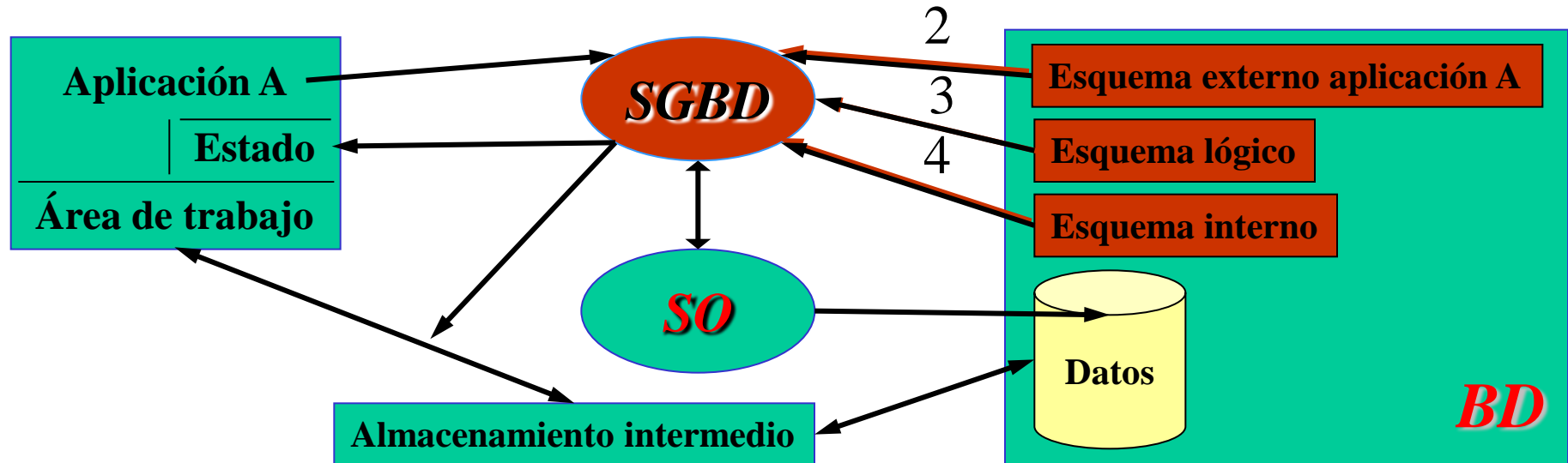
- ☹ un área de trabajo con sus áreas de entrada/salida y
- ☹ un área de estado destinada a recibir los mensajes y la información de control

➤ *Paso 1:* La aplicación hace una llamada al SGBD



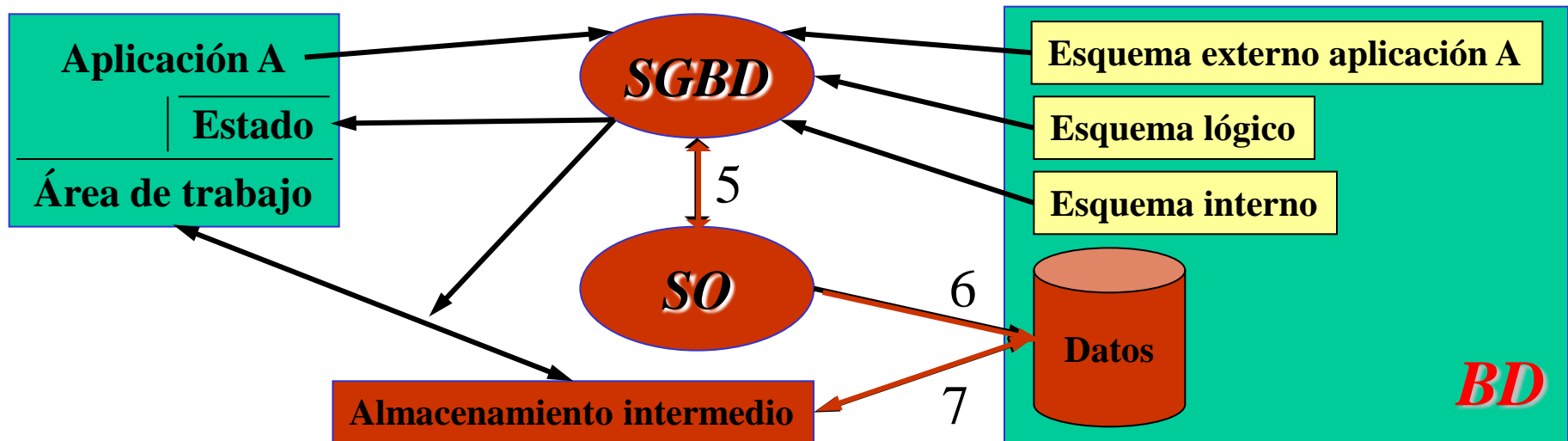
# Interacción del SGBD con el SO

- *Paso 2:* El SGBD obtiene el esquema externo de la aplicación A y examina la descripción de los datos solicitados
- *Paso 3:* El SGBD obtiene el esquema lógico y realiza la transformación externo/lógica correspondiente
- *Paso 4:* El SGBD examina el esquema interno y realiza la transformación lógico/física correspondiente



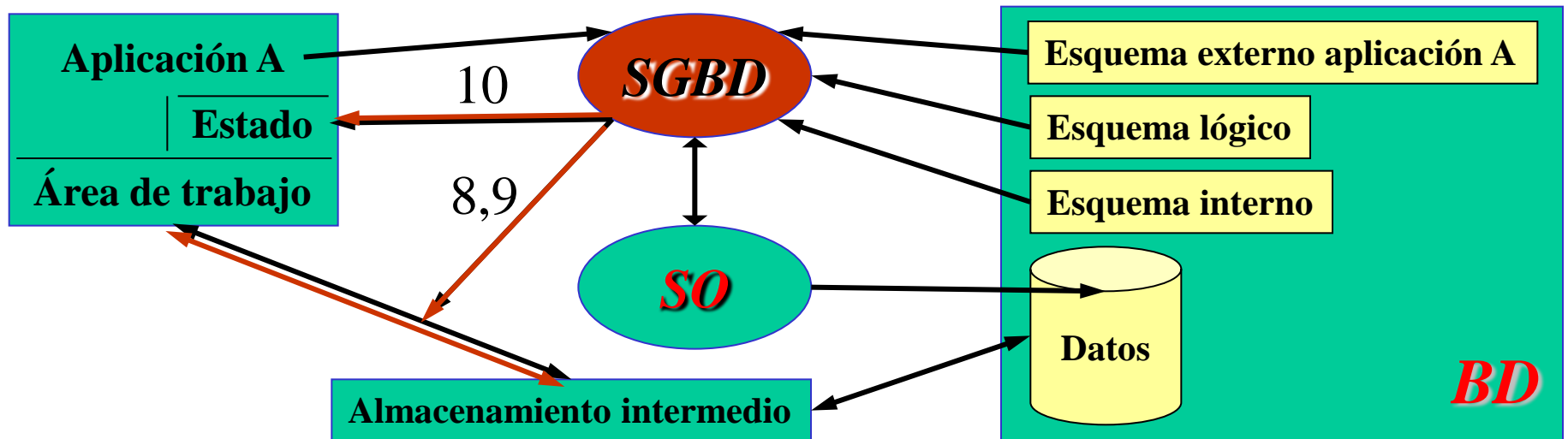
# Interacción del SGBD con el SO

- *Paso 5:* El SGBD emite órdenes al SO indicando que debe leer y donde está almacenado
- *Paso 6:* El SO interactúa con el almacén físico en el que se encuentran los datos
- *Paso 7:* Los datos se transfieren al almacenamiento intermedio (buffers)



## Interacción del SGBD con el SO

- *Paso 8:* El SGBD, comparando el esquema externo de la aplicación A y el esquema lógico, deduce los datos pedidos por el programa y lleva a cabo las transformaciones necesarias
- *Paso 9:* El SGBD transfiere los datos al área de trabajo
- *Paso 10:* El SGBD suministra información sobre el estado de la petición, indicando cualquier posible error





# Modelos de datos

# Modelo de datos

- Un modelo es una **herramienta** intelectual
  - & conjunto de conceptos y reglas
  - que permite representar**
    - & con diferentes niveles de abstracción
  - los aspectos *estáticos* y *dinámicos* de la parcela del mundo real que es objeto de estudio**
- Normalmente, se desea representar la **estructura de una base de datos**
  - ☹ y dicha estructura recibe el nombre de *esquema*



Grupo de Estructuras de Datos

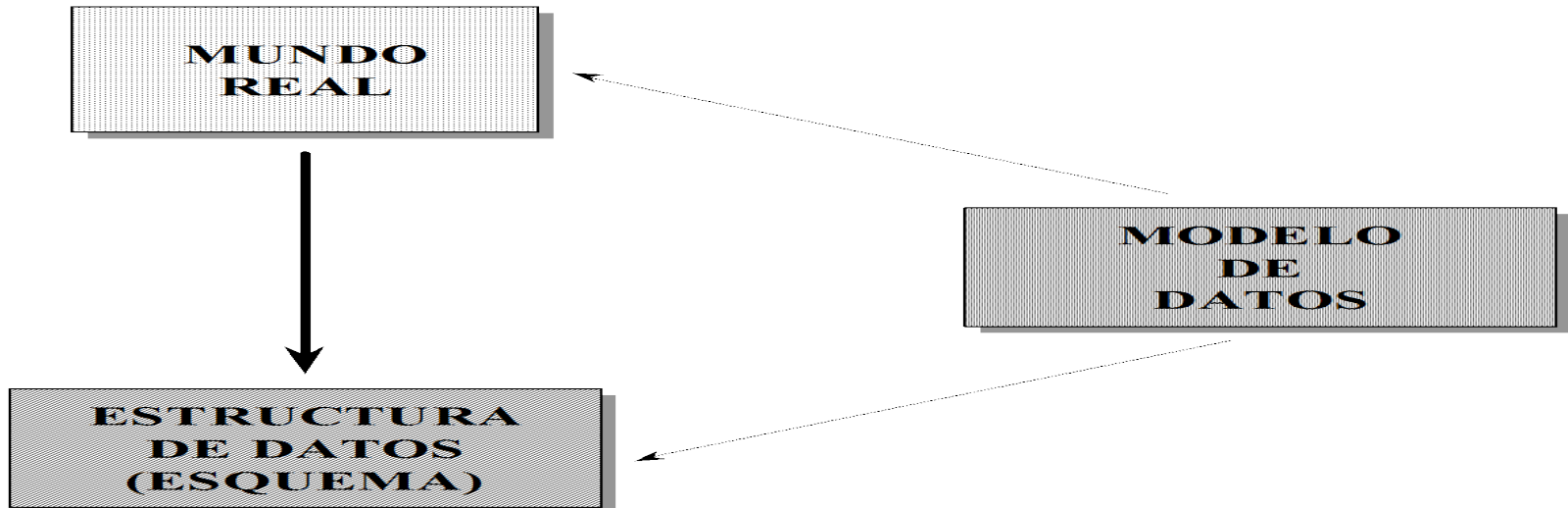
# Modelo .vs. esquema

✍ El **modelo** de datos es el instrumento que **se aplica a los datos** para **obtener el esquema**

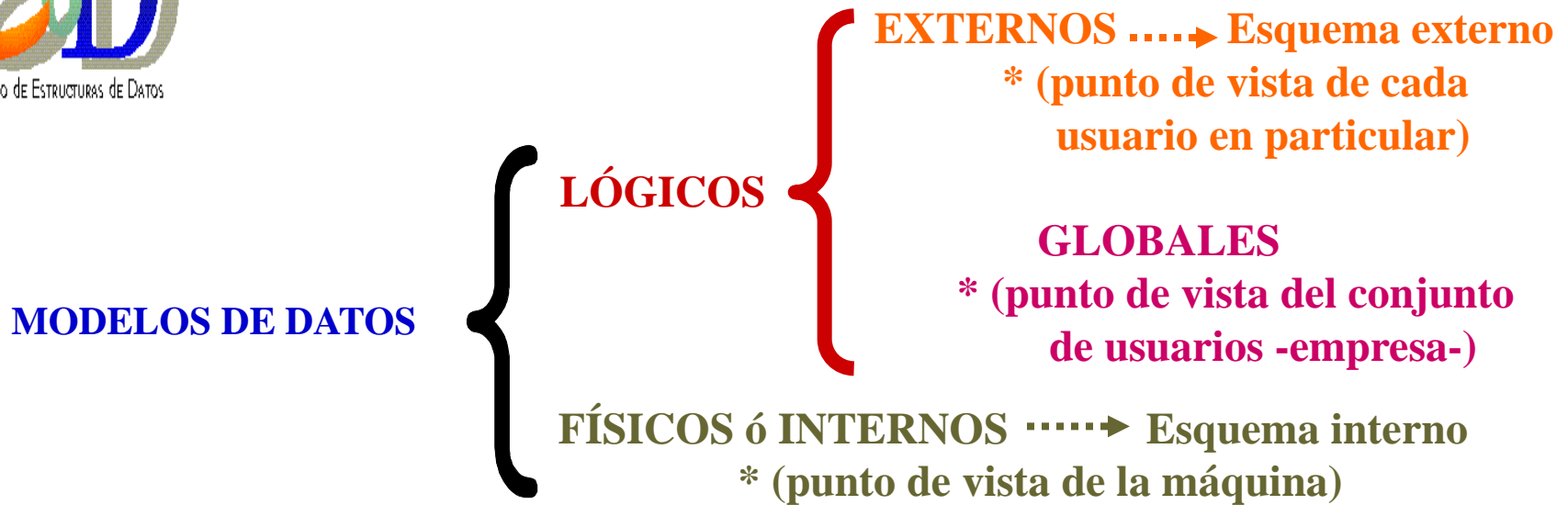
☹ Esta **distinción** entre modelo y esquema es **importante**

↩ & desafortunadamente, no se hace en la mayor parte de los textos  
dándose el nombre de modelo tanto al propio instrumento como al esquema resultante de su aplicación

- lo que puede dar lugar a confusiones



# Clasificación de los modelos de datos



✎ Nos vamos a **centrar** en los **modelos globales**

☹ ya que los **externos** suelen utilizar los **mismos conceptos** que los correspondientes **globales**

☹ y los **internos** ni están **estandarizados**, ni existen en realidad como tales modelos

& sino que son propios de cada uno de los productos comerciales

✎ Corresponden a lo que, en la arquitectura ANSI, se conoce como nivel conceptual

# Clasificación de los modelos de datos globales



# Modelo conceptual .vs. Modelo convencional

## CONVENCIONALES

## CONCEPTUALES

- |   |   |   |
|---|---|---|
| - Implementados en SGBD comerciales                     | ↔ | - No suelen estar implementados en SGBD |
| - Dependen del SGBD                                     | ↔ | - Independientes del SGBD               |
| - Más próximos al ordenador                             | ↔ | - Mayor nivel de abstracción            |
| - Poca capacidad semántica                              | ↔ | - Mayor capacidad semántica             |
| - Más enfocados a la implementación                     | ↔ | - Más enfocados al diseño de alto nivel |
| - Interfaz informático/sistema                          | ↔ | (modelado conceptual)-                  |
| - Nivel de “mediación” entre el nivel externo e interno | ↔ | - Interfaz usuario/informático          |

# Esquema .vs. ocurrencia

✍ Es preciso distinguir entre *esquema*

& descripción de la estructura de la base de datos

y *ocurrencia* del esquema

& los datos que en un determinado momento se encuentran almacenados en el esquema

☹ El *esquema* **no varía mientras no cambie** el *mundo real* que éste describe

☹ Una **ocurrencia** del esquema, puede **variar con** el transcurso del **tiempo**

# Esquema .vs. ocurrencia

☹ Al igual que en los lenguajes de programación existen *variables* de un cierto **tipo**, las cuales tienen en un momento determinado cierto **valor**



en las bases de datos se debería hablar de *variables de bases de datos*, cuyo **tipo** sería el **esquema** y cuyo **valor**, en un momento determinado, sería una **ocurrencia** del esquema

☹ Utilizaremos el termino *ocurrencia*, por ser el más extendido

& aunque también se usa a veces *instancia* cuyo significado según el DRAE no responde en absoluto a lo que aquí se trata de expresar

- *ejemplar, realización o estado* son vocablos mucho más apropiados pero muy poco utilizados



# Estática .vs. dinámica

✎ Las propiedades del MD son de dos tipos:

☹ **estáticas: relativamente invariantes en el tiempo**

& responden a lo que se suele entender como **estructura**

& y se representan en el **esquema**

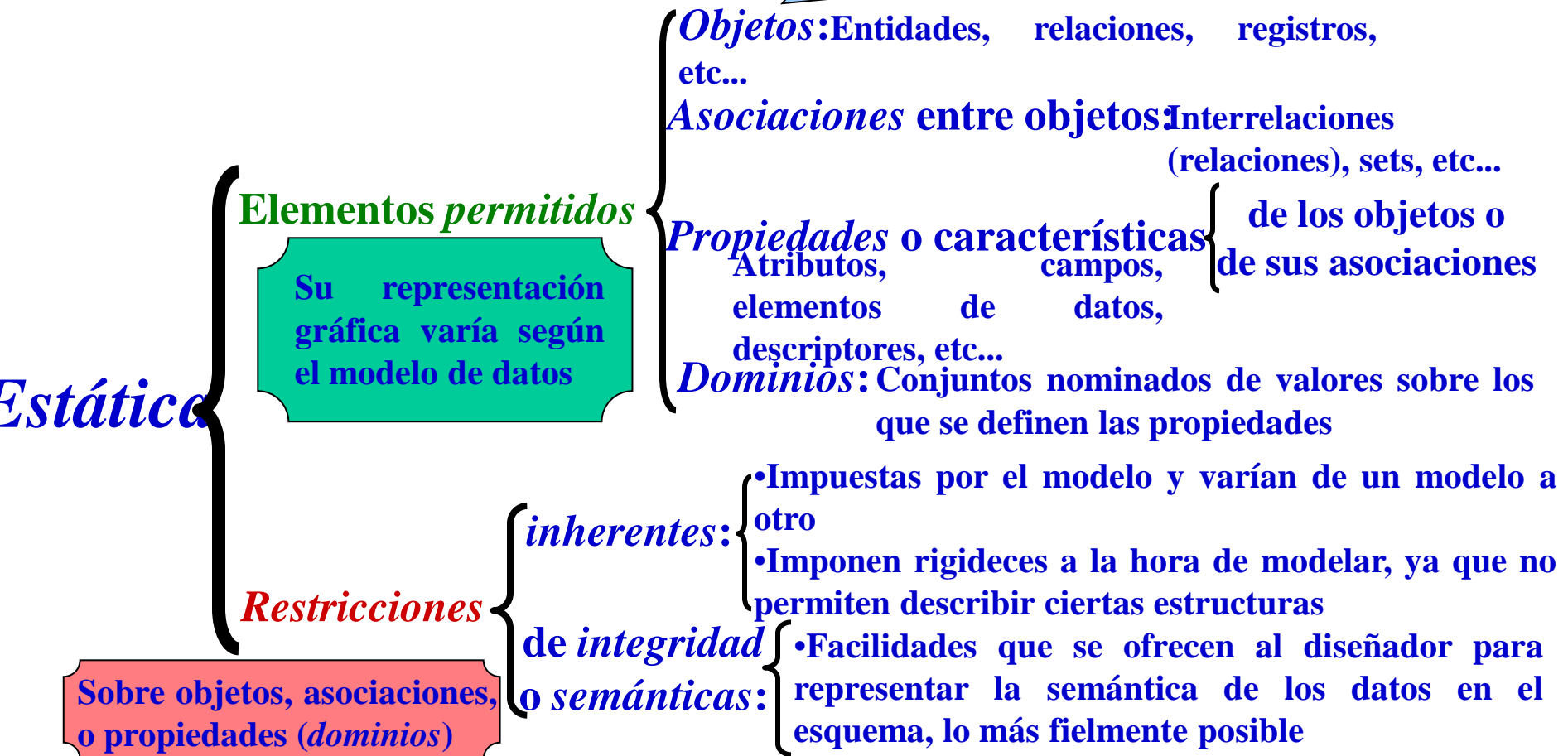
☹ **dinámicas:** son las **operaciones** que se aplican a los **datos** almacenados en las estructuras

● los **datos varían** en el transcurso del tiempo al aplicárseles dichas **operaciones**

& Estas propiedades se describen en las **especificaciones de transacciones**

# Estática de los modelos de datos

El término objeto tiene aquí la acepción del lenguaje común, no el significado específico que se le da cuando se estudia el paradigma de la orientación al objeto



# Ejemplos de elementos de un modelo

☹ En una empresa existen:

& **Objetos** como *empleados y departamentos*

- **Propiedades de los objetos:**

- \* Empleados: *nombre, apellidos, fecha de nacimiento, dni, ...*

- \* Departamentos: *código, nombre, dni del director, fecha de toma de posesión...*

& **Asociaciones** entre ellos: *asignaciones de empleados a departamentos, ¿ven más?...*

- **Propiedades de las asociaciones:** *horario del empleado en el departamento, ...*

& Las propiedades de fecha estarán definidas sobre el **dominio** de las fechas válidas

- \* no servirá el 32/9/78 ni el 23/14/89

- valga como ejemplo de restricción semántica de dominio sobre las propiedades

- El concepto de **dominio** es muy importante porque **permite restringir** las **comparaciones** permitidas entre propiedades

- \* **no son comparables si no están definidos en el mismo dominio**

# Ejemplos de elementos de un modelo

& No todas las restricciones semánticas sobre las propiedades han de ser de dominio:

- no debe permitirse la existencia de empleados que no tengan nombre (**valor nulo de la propiedad**)

& Se pueden emitir **restricciones semánticas sobre los objetos**:

- *No pueden existir dos empleados con el mismo dni*

& y sobre las **asociaciones**:

- *un empleado no puede trabajar en dos departamentos con horarios solapados*
- *todo departamento debe tener al menos un empleado*

# Dinámica de los modelos de datos



# Validación de ocurrencias del esquema

Sean  $BD_i$  y  $BD_j$  las ocurrencias del esquema o estados de la base de datos en los tiempos  $t_i$  y  $t_j$

Si entre  $t_i$  y  $t_j$  se ha producido un cambio en algún valor de la base de datos

⊖ alta, baja o modificación

**$BD_i \neq BD_j$**

Tanto  $BD_i$  como  $BD_j$  deben ser ocurrencias válidas de la base de datos

⊖ deben cumplir las *restricciones de integridad*

⊖ y también se deben cumplir las posibles *restricciones dinámicas* asociadas a los cambios de estado

# Operadores

↪ La componente **dinámica** del **modelo** consta de un conjunto de ***operadores***

☹ **definidos** sobre la **estructura** del correspondiente **modelo** de datos

& ya que **no todas** las **estructuras** **admiten** el mismo **tipo** de **operaciones**

↪ La **aplicación de una operación** a una ocurrencia de un esquema **transforma** a ésta en otra **ocurrencia**:

$$O(BD_i) = BD_j$$

☹ Pudiendo ser  $BD_i = BD_j$

& por ejemplo:

- en caso de consulta
- o cuando falla una operación por haberse producido un error

# Uso de indicadores

& Si consideramos que el **estado** de la base de datos viene **determinado** no sólo **por** los **valores** que toman los objetos del esquema

- sino **también** por los valores de sus *indicadores*

✱ por ejemplo el *indicador de error*

**cualquier operación hace variar el estado de la BD**

- **bien porque cambian los valores de los objetos**

✱ **en caso de una actualización**

- **bien porque cambian los indicadores**

✱ **en caso de fallo o de consulta**

& En algunos MD como el Codasyl la manipulación de los datos está basada en los indicadores

 **datos**



# Localización y acción

✍ Una **operación** tiene **dos componentes**:

☹ **Localización** o *enfoque* o *selección*

& consiste en localizar

- una **ocurrencia** de un **objeto** indicando un **camino** (*navegacional*)
- o un **conjunto de ocurrencias** especificando una **condición** (*especificación*)


☹ **Acción**

& que se realiza sobre la(s) ocurrencia(s) previamente localizada(s) y que puede consistir en una

- *recuperación*
- o una *actualización*

# Localización y acción

✎ La **distinción** entre localización y acción es de tipo **formal**

☹ si bien algunos lenguajes  


- como el LMD de **Codasyl**

**tienen dos mandatos distintos**

& uno para expresar la *selección*

& y otro para la *acción*

☹ otros lenguajes  


- como el SQL

**reúnen ambas operaciones en un único operador**

# Localización y acción

✎ Sin seguir una sintaxis concreta, sino más bien en un plano **conceptual**, podemos expresar una sentencia del LMD de la siguiente forma:

**LOCALIZACION** <condición>

**ACCION** <objetivo>

- donde LOCALIZACION y ACCION son mandatos del LMD
- <condición>
  - ✳ representa una **expresión lógica** proporcionada por el usuario **que deben cumplir** los **objetos** que se desea **localizar**
  - ✳ o especifica el **camino** que indica el usuario **para llegar** a esos **objetos**
- mientras que <objetivo> son los **objetos** (o las propiedades de éstos) **sobre** los que el usuario desea que se **aplique** la **acción**

# Evitar el esquema externo

## ACCION <objetivo>

- <objetivo> son los **objetos** (o las propiedades de éstos) sobre los que el usuario desea que se **aplique** la **acción**

☹ Si el **SGBD** se adaptase **estrictamente** a la arquitectura a tres niveles de ANSI

- <objetivo> debería ser el **nombre** de un **esquema externo** previamente definido

& **sin embargo**, algunos SGBD, especialmente los basados en el modelo relacional

- **no** obligan a definir previamente el *esquema externo*

& **permitiendo** describir el **objetivo** dentro de la misma **sentencia** de manipulación

# Localización y acción en un mandato, evitando el esquema externo

- ☹ Como ejemplo, en la siguiente sentencia SQL se especifican conjuntamente <objetivo> y <condición>

```
SELECT Título, Autor  
FROM Libro  
WHERE Fecha_Edicion = "1996"
```

& La localización y la acción



- en este caso, **recuperar**

se expresan mediante un único mandato con el verbo inglés ***SELECT***

& el objetivo son las propiedades (*atributos* en el modelo relacional) *Título y Autor* del objeto (*relación*) *Libro*

& y la condición es que la *fecha de edición* del libro sea igual a *1996*

- ☹ No se hace referencia a ningún esquema externo (*vista*)

& ya que la estructura objetivo que se desea recuperar (*Título y Autor* de **LIBRO**) se incluye en la misma sentencia

# Restricciones de integridad

## Manual-Programática-Integrada



```

.....
IF SALARIO > 1 M
THEN ERROR
.....
IF EMPLEADO.DEPT = 'NULL'
THEN ERROR
.....
    
```



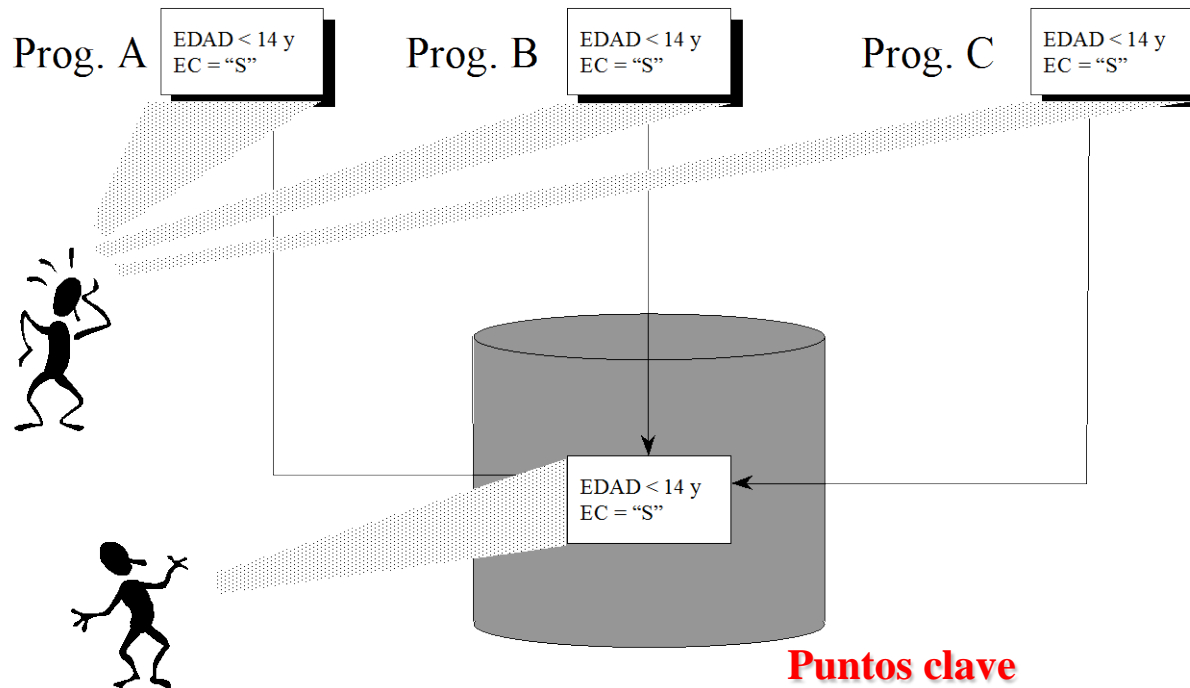
**USUARIO**

**PROGRAMAS**

**BASE DE DATOS**

**MIGRACION DE LA SEMANTICA**


# Semántica dispersa .vs. Semántica integrada



- ⌚ Carga de programación (redundancia de especificación de reglas)
- ⌚ Posibilidad de inconsistencia (no a nivel de datos sino de reglas)
- ⌚ Accesibilidad del significado de la información
- ⌚ Tener siempre en mente que las restricciones semánticas no reconocidas por el modelo han de ser:

- II documentadas durante el diseño, a fin de que sean tenidas en cuenta siempre
- II incluidas en la programación, con los inconvenientes que esto conlleva

# Evolución de los modelos de datos: modelos de datos primitivos

 Aparte de la clasificación de los modelos actuales, ya presentada se puede estudiar la evolución que han realizado a través de tres generaciones

## Modelos de datos *primitivos*:

- ☹ Coinciden con los sistemas de gestión de ficheros que soportaron los SI durante los años 60 y parte de los 70
  - & Los *objetos* se representan como registros organizados en ficheros
  - & y las interrelaciones entre objetos a través de referencias explícitas
    - campo del registro cuyo valor referencia a otro objeto
- ☹ Los lenguajes de manipulación son totalmente dependientes de la organización física de los ficheros
  - métodos de acceso
  - operaciones básicas primitivas para ficheros: lectura, escritura y poco más.



# Evolución de los modelos de datos: modelos de datos clásicos

## Modelos de datos clásicos:

☹ Los modelos *jerárquico* y *red* aparecieron como **extensiones** de los modelos **primitivos**

- buscando **mayor eficiencia** en la **manipulación** de las **asociaciones** entre objetos

& Los **objetos** se siguen representando como **registros organizados en ficheros**

& Pero proporcionan **estructuras de datos de referencias explícitas más complejas**

\* *árbol*

\* *set*

que permiten expresar directamente las interrelaciones entre objetos (registros)

- Sus lenguajes de manipulación contienen operadores para manejar estas estructuras

# Evolución de los modelos de datos: modelos de datos clásicos

- ☹ El modelo *relacional* supuso una ruptura con la situación anterior
  - & Tanto los **objetos** como las **interrelaciones** entre los mismos se representan mediante **tablas** (*relaciones*)
  - & Aparecen los **lenguajes de manipulación declarativos**

# Comparativa resumen modelos clásicos: Sencillez

- ✎ La sencillez es una característica del modelo relacional que lo hace muy asequible a los usuarios
  - ☹ sencillez en la estructura por uniformidad
  - ☹ y sencillez correspondientemente en las operaciones de consulta y de actualización
- ✎ Esta característica se diluye en los modelos jerárquico y Codasyl
  - ☹ debido a que desaparece la uniformidad de representación estructural

# Comparativa resumen modelos clásicos: Independencia físico/lógica

✎ El modelo Codasyl, y aún más el Jerárquico, presenta una correspondencia directa entre las relaciones lógicas y los caminos de acceso físicos

☹ lo que dificulta la independencia

& como consecuencia, los lenguajes asociados son de tipo navegacional

✎ En el modelo Relacional el acceso a los datos se realiza en función de sus propiedades

☹ el usuario no conoce los caminos de acceso

& el sistema es el que se ocupa de seleccionar el camino físico

- optimizando los recursos y el tiempo de respuesta

& se utilizan lenguajes de especificación

# Comparativa resumen modelos clásicos: Rendimiento

- ✎ Ha sido un argumento fundamental para los que defendían la supremacía de los sistemas tipo red
  - & Sin embargo, CODD y DATE siempre han sostenido que no había razones objetivas para que los sistemas relacionales no fuesen tan eficientes como los basados en otros modelos
- ☹ El rendimiento de las actuales versiones de algunos SGBDR parece que viene a dar la razón a estos autores
  - & Fundamentalmente debido a avances en la optimización

# Evolución de los modelos de datos: modelos de datos semánticos

## Modelos de datos semánticos:

- ☹ Surgen con la intención de aumentar la capacidad expresiva de los modelos clásicos
  - & Incorporan conceptos y mecanismos de abstracción que permiten modelar la realidad de una forma más natural
- ☹ Salvo excepciones han sido usados fundamentalmente como herramientas para el diseño conceptual

# Los modelos de datos en el diseño de bases de datos

