CONCEPTOS DE BASES DE DATOS

CLASE 1

Bibliografía

- **Introducción a las Bases de Datos. Fundamentos y Diseño. *(Bertone - Thomas)***
- Files & Databases: An Introduction *(Smith-Barnes)*
- Estructuras de Archivos *(Folk-Zoellick)*
- Fundamento de sistemas de BD *(Elmasri - Navathe)*
- Fundamentos de Bases de Datos *(Korth Silvershatz)*

Conceptos básicos

- **Base de datos (BD)**
  - Cualquier información dispuesta de **manera adecuada** para su tratamiento por una computadora
  - Colección de archivos diseñados para servir a **múltiples aplicaciones**
  - Conjunto de **datos interrelacionados** con un propósito específico vinculado a la resolución de un problema del mundo real

Conceptos básicos

- **Orígenes de las BD**
  - Aplicaciones antiguas
    - Usaban archivos propios
    - Consultaban uno o más archivos maestros → acceso secuencial
    - Actualizaban uno o más archivos maestros → acceso secuencial

Conceptos básicos

- **Orígenes de las BD**
  - Aplicaciones antiguas
    - Añadir una nueva aplicación requería datos que ya existían y datos nuevos → nuevo archivo propio
    - Menor competencia por el acceso a los archivos maestro
    - Repetición de datos

Conceptos básicos

- **Orígenes de las BD**
  - Aplicaciones antiguas
    - La tecnología fue avanzando (redes, servidores, equipos terminales, PC, discos locales) → **los sistemas de información evolucionan**
    - Los requerimientos de las aplicaciones cambian
    - Las aplicaciones se integran:
      - Interrelación de los archivos
      - Eliminación de la redundancia de datos

Conceptos básicos

- **SGBD** (DBMS)
  - "Sistema de Gestión de Bases de Datos" o "Data Base Management System"
  - Es un sistema de software (colección de programas) que permite a los usuarios crear y mantener la BD
    - Facilita los procesos de definición, construcción y manipulación de la BD

Conceptos básicos

- **SGBD** → Funcionalidad
  - Control de redundancia
  - Acceso a los datos en todo momento
  - Acceso concurrente a los datos
  - Seguridad: control de acceso a datos, usuarios, recursos, backups, entre otros.
  - Integridad: persistencia de datos aún ante fallos, restricciones de datos, etc.

Conceptos básicos

- **Propósitos de la asignatura**
  - Estudio de archivos
    - Algorítmica clásica de archivos
    - Archivos de datos y archivos de acceso a datos
    - Alternativas de acceso a archivos con bajo costo

Archivos

- **Definiciones**
  - Un archivo es una colección de registros semejantes, guardados en dispositivos de almacenamiento secundario de la computadora
  - Un archivo es una estructura de datos que recopila, en un dispositivo de almacenamiento secundario de una computadora, una colección de elementos del mismo tipo
  - Un archivo es una colección de registros que abarcan entidades con un aspecto común y originadas para algún propósito particular

Archivos

- **Organización interna de los datos**
  - Secuencia de bytes
  - Campos
  - Registros

Archivos

- **Organización interna de los datos**

  - Secuencia de bytes

    - Se determina como **unidad más pequeña de L/E** al byte.
    - No se puede determinar fácilmente el comienzo y el final de cada dato
    - Generalmente son archivos de texto
    - Lecturas y escrituras → procesos ad-hoc

Archivos

- **Organización interna de los datos**
  - Campos
    - Se determina como **unidad más pequeña de L/E** al campo.
    - El campo es un ítem de datos elemental y se caracteriza por su tipo de dato y su tamaño, por ejemplo:
      - Nro. Entero (2 bytes)
      - Nro. flotante (6 bytes)
      - Carácter (1 byte)
      - Cadena de caracteres (256 bytes)
    - Las lecturas y escrituras se hacen a nivel campo.

Archivos

- **Organización interna de los datos**

  - Registros

    - Se determina como **unidad más pequeña de L/E** al registro.
    - El registro es un conjunto de campos agrupados que definen un elemento del archivo.
    - Los campos internos a un registro deben estar lógicamente relacionados, como para ser tratados como una unidad.
    - Las lecturas y escrituras se hacen a nivel registro.

Archivos

- **Acceso a los datos**
  - Secuencial
  - Secuencial indizado
  - Directo

Archivos

- **Acceso a los datos**
  - Secuencial
    - Los soportes de datos secuenciales son aquellos en los que los datos están escritos **unos a continuación de otros**.
    - Por ejemplo, para acceder a un determinado registro, se necesita pasar por todos los registros anteriores, según el orden físico en el que están guardados

Archivos

- **Acceso a los datos**

  - Secuencial indizada

    - El tipo de acceso a los datos es secuencial, pero **siguiendo el orden establecido por otra estructura**.
    - En este caso la secuencia a seguir está determinada **lógicamente**, y no por el orden físico en el que están guardados los elementos.
    - Ejemplo: archivo con índice

Archivos

- **Acceso a los datos**

  - Directo

    - Los soportes de datos direccionables se estructuran de modo que los elementos de datos pueden ser localizados directamente por su dirección, y no se requiere pasar por los registros anteriores.

    - Para hacer posible el acceso directo, los elementos deben tener un campo clave unívoco, que los diferencie del resto de los elementos del archivo.

Archivos

- **Tipo de almacenamiento**
  - Primario
  - Secundario

Archivos

- **Tipo de almacenamiento**
  - Primario → RAM
    - Capacidad de almacenamiento limitada.
    - Volátil.
    - Alto costo.
    - Acceso rápido (orden de nanosegundos)

Archivos

- **Tipo de almacenamiento**
  - Secundario → Cintas y discos
    - Alta capacidad de almacenamiento.
    - No volátil
    - Menor costo que el almacenamiento primario.
    - Acceso “lento” (orden de milisegundos) → se debe optimizar
      - Búsqueda de un único dato → obtención en un intento, o en pocos
      - Búsqueda de varios datos → obtención de todos de una sola vez

Archivos

- **Tipo de almacenamiento**

  - Secundario → Cintas y discos

    - Cintas → acceso secuencial. Medio económico, estables en diferentes condiciones ambientales y fáciles de transportar

    - Discos → acceso directo. Se almacenan los datos en sectores:

      - Un registro en un solo sector

      Ventaja: cualquier registro se recupera con sólo recuperar un sector.
      Desventaja: puede quedar espacio sin uso

      - El principio de un registro en un sector y el final en otro

      Ventaja: se evita que quede espacio sin uso
      Desventaja: acceso a dos sectores en vez de uno

Archivos

- **Niveles de visión**
  - Física
    - Archivo que existe en el almacenamiento secundario.
    - Es conocido por el S.O. y aparece en su directorio.
  - Lógica
    - Visto desde dentro del programa.
    - Independencia física: se realizan operaciones básicas sobre los archivos sin conocer su ubicación física real.

Archivos

Camino de los datos

- **Camino de los datos** → no es trivial
  - Al escribir un dato en un archivo desde un programa, intervienen varios componentes del SO:
    - Administrador de archivos
    - Buffer de E/S
    - Procesador de E/S
    - Controlador de disco

Archivos

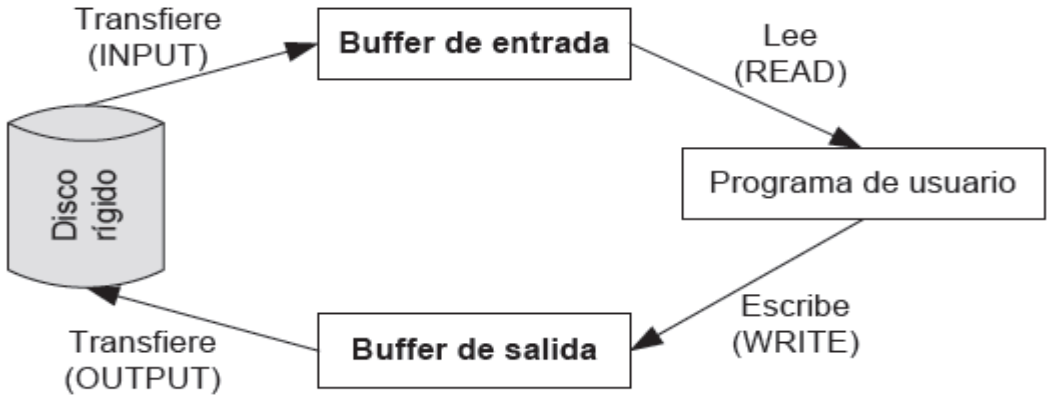
Camino de los datos

- **Administrador de archivos**
  - Conjunto de programas del S.O. que tratan aspectos relacionados con archivos y dispositivos de E/S
  - Capas superiores (aspectos lógicos → tabla)
    - Establecer si las características del archivo son compatibles con la operación deseada (abierto/cerrado, tipo archivo, tipo operación, etc.)
  - Capas inferiores (aspectos físicos)
    - Determinar donde se guarda el dato (cilindro, pista, sector)
    - Si el sector está ubicado en RAM se utiliza, caso contrario debe traerse previamente.

Archivos

Camino de los datos

- **Buffers de E/S**
  - Agilizan la E/S de datos (reducen el acceso a almacenamiento sec.)
  - Memoria intermedia entre un archivo y un programa, donde los datos residen provisoriamente:
    - Al escribirlos, hasta ser almacenados en forma definitiva en memoria secundaria
    - Al leerlos, una vez recuperados desde la memoria secundaria
  - Los buffers ocupan lugar en RAM y son manipulados por el S.O.



Archivos

Camino de los datos

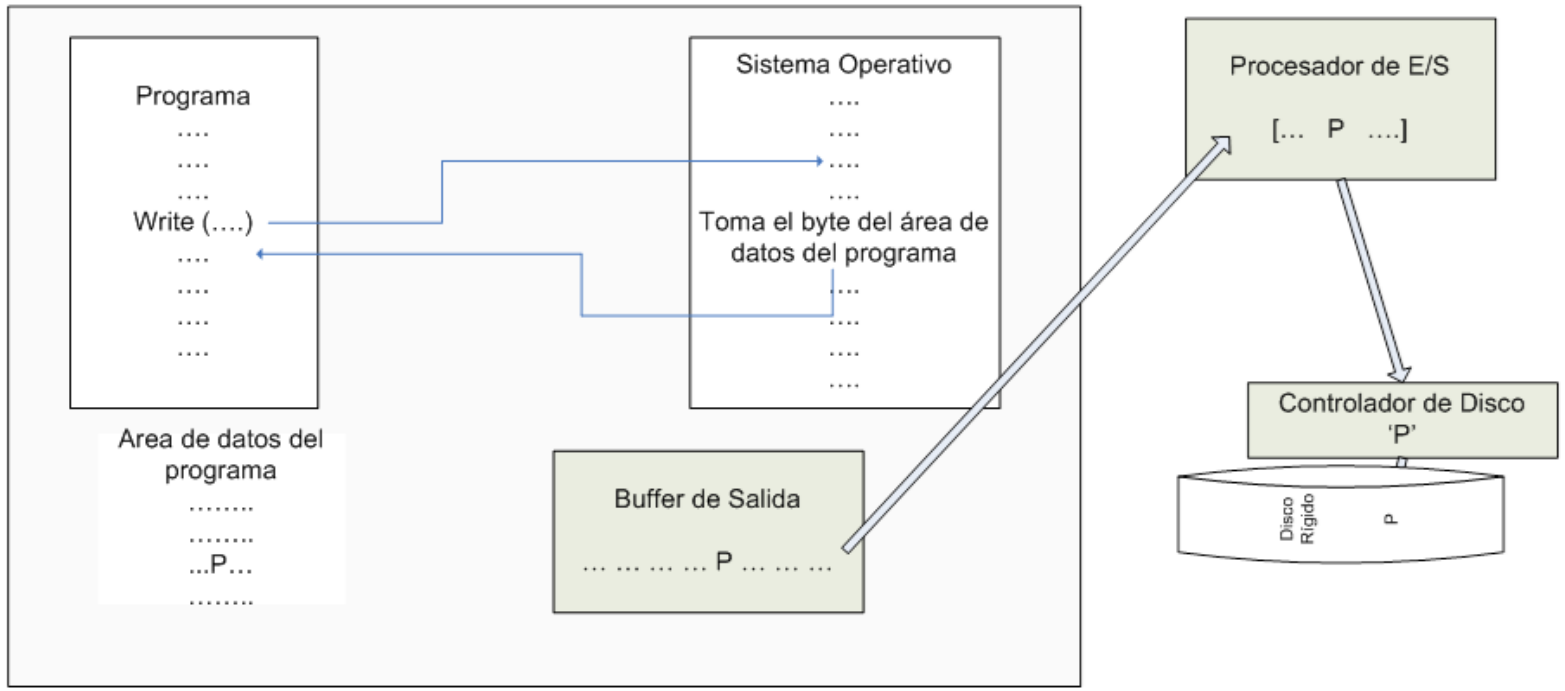
- **Procesador de E/S**
  - Accede a los buffers de E/S y envía los requerimientos de lectura/escritura al controlador de disco
    - Es independiente de la CPU
- **Controlador de disco**
  - Encargado de controlar la operación de disco
    - Colocarse en la pista correspondiente
    - Colocarse en el sector correspondiente
    - Transferir los bytes hacia/desde disco

Archivos

Camino de los datos

Tabla

Nombre	Abierto por	Acceso	Propietario	Protección
archivo1	Perez	L/E	Gomez	prop:L/E otro: L/E
archivo2	García	L	García	prop:L/E otro: L
TEXTO	Gomez	E	Gomez	prop:L/E otro: E



Archivos

Operaciones básicas

- Notación del lenguaje Pascal
- Declaración
  - **Var** miArchivo: **file of** *tipo_de_dato*;
  - **Type** tipoArchivo: **file of** *tipo_de_dato*;
    Var miArchivo: tipoArchivo;
- Ejemplo

```
Type persona = record
        nombre: string[20];
        dni: string[10];
        domicilio: string[40];
        edad: integer;
      end;
```

```
numeros = file of integer;
empleados = file of persona;

Var   fileNros: numeros;
      fileEmp: empleados;
```

Archivos

Operaciones básicas

- Relación con el SO: se debe establecer la correspondencia entre el nombre físico y el nombre lógico
  - **Assign**(nombreLogico, nombreFisico);
- Ejemplo

```
Var   fileNros: numeros;  fileEmp: empleados;
Begin
      …
      Assign(fileNros, ‘numeros.dat’);
      Assign(fileEmp, ‘empleados.dat’);
      …
```

Archivos

Operaciones básicas

- Apertura / Creación
  - **Rewrite**(nombreLogico); → solo escritura (creación)
  - **Reset**(nombreLogico); → lectura/escritura
- Cierre
  - **Close**(nombreLogico);
  - Se usa cuando no se va a trabajar más con el archivo.
  - Pone una marca de **EOF** (*End Of File*) al final del archivo.

En todos los casos ***nombreLogico*** es una variable de tipo archivo sobre la que se realizó la asignación correspondiente.

Archivos

Operaciones básicas

- Lectura / Escritura
  - **Read**(nombreLogico, variable);
  - **Write**(nombreLogico, variable);
  - Estas operaciones se realizan sobre los buffers E/S relacionados a los archivos.
  - En ambos casos la variable debe ser del mismo tipo que los elementos que se declararon como parte del archivo.

Archivos

Operaciones básicas

- Ej: creación de un archivo de números

```
Program generarArchivo;
type
    {tipo de dato del archivo}
    archivo = file of integer;
var
    {nombre lógico del archivo}
    fileLogico: archivo;
    {p/ obtener info de teclado}
    nro: integer;
    {p/ obtener nombre físico del archivo desde teclado}
    nombreFile: string[12];
begin
    {se pide y obtiene el nombre desde teclado}
    write('Ingrese el nombre del archivo:' );
    read(nombreFile);
    assign(fileLogico, nombreFile);
    {se crea el archivo}
    rewrite(fileLogico);
    {se obtiene de teclado el primer valor}
    read(nro);
    {se repite hasta leer un número igual a cero}
    while (nro <> 0)do
    begin
      {se escribe el nro en el archivo  y se lee uno nuevo}
      write(fileLogico, nro);
      read(nro);
    end;
    {se cierra el archivo}
    close(fileLogico);
end.
```

Archivos Operaciones adicionales

- Funciones
  - **EOF**(nombreLogico); → Fin de archivo
    - Hay que usarla antes de intentar leer desde un archivo.
  - **Filesize**(nombreLogico); → Tamaño
    - Devuelve la cantidad de registros del archivo.
  - **Filepos**(nombreLogico); → Posición
    - Devuelve la posicion actual en el archivo *(0..N-1)*
  - **Seek**(nombreLogico, pos); → Posicionamiento
    - Permite ir a una posicion determinada del archivo.

Archivos

Ejemplo

- Ej: presentación de un archivo en pantalla

```
Procedure presentarArchivo(var fileLogico: archivo);
var
    nro: integer;        {p/ leer elemento del archivo}
begin
    {se abre como L/E el archivo creado anteriormente}
    reset(fileLogico);
    {se repite mientras haya elementos en el archivo}
    while (not eof(fileLogico))do
    begin
            read(fileLogico, nro);  {se obtiene un elemento desde el archivo}
            writeln(nro);  {se presenta el elemento en pantalla}
    end;
  close(fileLogico);
 end;
```