



Fundamentos de Organización de Datos

Curso 2015

La cátedra

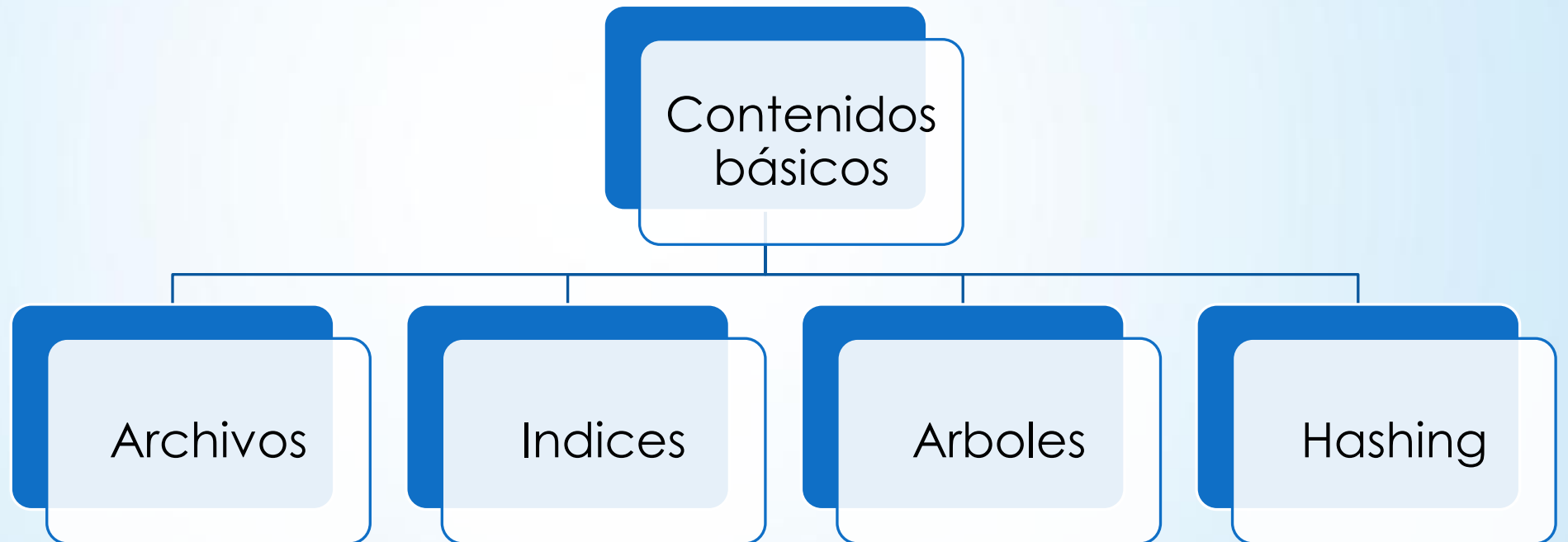
► Clases

- Teóricas
- Explicaciones de Prácticas (donde se presentan ejemplos)
- Prácticas
- Se utilizará la plataforma WebInfo

► Para aprobar la cursada

- Un Parcial
- Dos recuperatorios

La Materia



Calendario

4

Semana	Teoría	Práctica
02/03	Introducción. Archivos, Algorítmica Básica	Sin Actividad
09/03	Archivos, Algorítmica Básica, Algorítmica Clásica	Archivos Algorítmica Básica
16/03	Archivos, Algorítmica Clásica. Proceso de Baja en Archivos	Archivos Algorítmica Básica
23/03	Archivos con Registros de Longitud Variable	Sin actividad por feriado
30/03	Semana Santa	Archivos Algorítmica Clásica
07/04	Búsqueda de información en Archivos. Índices	Archivos Algorítmica Clásica
14/04	Arboles Binarios, AVL. Introducción a Arboles B	Bajas, Registros de Longitud Variable
21/04	Arboles B, Creación Búsqueda Eliminación, Performance	Bajas, Registros de Longitud Variable
28/04	Sin actividad por feriado	Arboles,
04/05	Arboles B * y B+	Arboles
11/05	Hashing	Arboles
18/05	Hashing	Hashing
25/05	Consulta	Hashing
01/06	Examen de Archivos	Consulta
08/06	Consulta	Primer Parcial (Martes 09/06/2015)
15/06	Examen de Arboles	Consulta
22/06	Consulta	Muestra de examen, consulta
29/06	Consulta	Recuperatorio (Martes 30/06/2015)
06/07	Examen de Hashing	Muestra de examen, consulta
13/07	Consulta	Recuperatorio (Martes 14/07/2015)
20/06	Vacaciones	Vacaciones
27/06	Vacaciones	Vacaciones
03/08		Muestra de examen. Cierre de cursada.

Propuesta de exámenes teóricos - Cursada 2015

- Podrán acceder a los exámenes escalonados todos los alumnos que se encuentren inscriptos en la asignatura.
- Se realizarán tres exámenes teóricos.
- La aprobación de cada examen es con nota 4 o superior
- Los exámenes NO tendrán recuperatorio.
- Para aprobar la parte teórica se deberán aprobar los tres exámenes.
- Se deberá aprobar la cursada durante el semestre en que rinde los exámenes teóricos.
- Deberá anotarse para pasar el final de la asignatura como máximo para la fecha de marzo de 2016.

IBD → FOD 2015

- Recursantes de IBD que se pasen a los planes 2015 y, que hayan rendido y aprobado el primer parcial de IBD en el año 2012 o posterior, se les dará aprobada la cursada de FOD.
- Deberán rendir el final de acuerdo al programa de FOD Plan 2015.

Bibliografía

- Introducción a las Bases de Datos. Conceptos Básicos (Bertone, Thomas)
 - Estructuras de Archivos (Folk-Zoellick)
 - Files & Databases: An Introduction (Smith-Barnes)
 - Fundamentos de Bases de Datos (Korth Silvershatz)

Fundamentos de Organización de Datos

Clase 1

Agenda

Conceptos básicos de BD

- Definiciones
- Características

Archivos

- Introducción
- Operatoria básica

Conceptos básicos

Qué es una Base de Datos?

Es una colección de datos relacionados.

Colección de **archivos** diseñados para servir a múltiples aplicaciones

Un dato representa hechos conocidos que pueden registrarse y que tienen un resultado implícito.

Conceptos básicos

Propiedad es implícitas de una BD:

Una BD representa algunos aspectos del mundo real, a veces denominado Universo de Discurso.

Una BD es una colección coherente de datos con significados inherentes. Un conjunto aleatorio de datos no puede considerarse una BD. O sea los datos deben tener cierta lógica.

Una BD se diseña, construye y completa de datos para un propósito específico. Está destinada a un grupo de usuarios concretos y tiene algunas aplicaciones preconcebidas en las cuales están interesados los usuarios

Una BD está sustentada físicamente en archivos en dispositivos de almacenamiento persistente de datos

Archivos

Definiciones

- Colección de registros guardados en almacenamiento secundario
- Colección de datos almacenados en dispositivos secundarios de memoria
- Colección de registros que abarcan entidades con un aspecto común y originadas para algún propósito particular

Archivos → algunos conceptos

Hardware

Almacenamiento
primario

Almacenamiento
secundario (DR)

Acceso a RAM
vs. Acceso a DR

Platos

Superficies

Pistas

Sectores

Cilindros

Comparaciones

Archivos → organización

Secuencia de bytes

- no se puede determinar fácilmente comienzo y fin de cada dato.
- Archivos de texto

Registros y Campos

- Campo: Unidad más pequeña, lógicamente significativa de un archivo
- Registros: Conjunto de campos agrupados que definen un elemento del archivo

Archivos → Acceso

Secuencial Físico: acceso a los registros uno tras otro y en el orden físico en el que están guardados

Secuencial indizado (lógico): acceso a los registros de acuerdo al orden establecido por otra estructura

- Ej: una guía telefónica, o índice temático del un libro

Directo: se accede a un registro determinado sin necesidad de haber accedido a los predecesores

De acuerdo a la forma de acceso

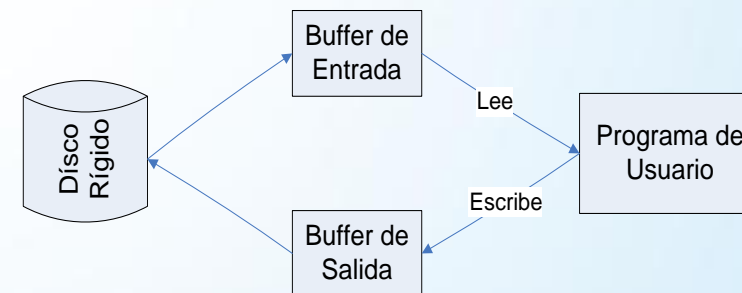
- **Serie** cada registro es accesible solo luego de procesar su antecesor, simples de acceder (*acceso secuencial físico*)
- **Secuencial** los registros son accesibles en orden de alguna clave (*acceso secuencial lógico*)
- **Directo** se accede al registro deseado (*acceso directo*)

Archivos

► Buffers

- Memoria intermedia entre un archivo y un programa, donde los datos residen provisoriamente hasta ser almacenados definitivamente en memoria secundaria o donde los datos residen una vez recuperados de dicha memoria secundaria.

- Los buffers ocupan lugar en RAM
- SO encargado de manipular los buffers
- Como trabaja?



Archivos → Operaciones básicas

Dos niveles

- Físico (almacenamiento secundario)
- Lógico (dentro del programa)
 - Operaciones
 - Crear
 - Abrir
 - Read/Write
 - Eof
 - Seek(localización)

Archivos → Declaraciones

Utilizaremos la notación Pascal

Declaración de archivos

- Variable
 - `Var archivo: file of Tipo_de_dato;`
- Tipo
 - `Type archivo: file of Tipo_de_dato;`
 - `Var arch: archivo`

Archivos – Operaciones Básicas

► Ejemplos

```
Type emple = record
    nombre: string [20];
    direccion: string [40];
    edad: integer;
end;
numero = file of integer;
empleado = file of emple;

Var arch_num: numero;
Var arch_emp: empleado,
```

Archivos → Operaciones Básicas

► Relación con el sistema operativo

- Se tiene que establecer la correspondencia entre el nombre físico y nombre lógico

Assign (n_lógico, N_físico)

► Ejemplo

```
Program archivos;  
...  
Type emple = record  
    nombre: string [20];  
    direccion: string [40];  
    edad: integer;  
end;  
numero = file of integer;  
empleado = file of emple;  
Var arch_num:numero;  
Var arch_emp:empleado;  
...  
Begin  
    ...  
    Assign( arch_num, 'pepe.dat' )  
    Assign( arch_emp, 'pipo.dat' )  
    ...  
End;
```

Archivos – Operaciones Básicas

```
Rewrite (nombre_logico);
```

- De solo escritura (creación)

```
Reset (nombre_logico);
```

- Lectura Escritura (apertura)

Nombre lógico representa una variable de tipo archivo sobre la que se realizó la asignación.

```
Close(nombre_logico);
```

- Cierre de archivo
- Esta instrucción indica que no se va a trabajar más con el archivo. Significa poner una marca de EOF (end of file) al final del mismo.

Archivos → Operaciones Básicas

```
Read(nombre_logico, variable);
```

```
Write(nombre_logico, variable);
```

Estas operaciones leen y/o escriben sobre los buffers relacionados a los archivos

No se realizan directamente sobre el DR

En ambos casos la variable debe ser del mismo tipo que los elementos que se declararon como parte del archivo

Archivos → EJ 1 Crear un archivo

```
Program Generar_Archivo;
  type archivo = file of integer; {definición del tipo de dato para el archivo }
  var arc_logico: archivo;         {variable que define el nombre lógico del archivo}
      nro: integer;                {nro será utilizada para obtener la información de teclado}
      arc_fisico: string[12];      {utilizada para obtener el nombre físico del archivo desde teclado}
begin
  write( 'Ingrese el nombre del archivo:' );
  read( arc_fisico ); { se obtiene el nombre del archivo}
  assign( arc_logico, arc_fisico );
  rewrite( arc_logico ); { se crea el archivo }
  read( nro ); { se obtiene de teclado el primer valor }
  while nro <> 0 do begin
    write( arc_logico, nro ); { se escribe en el archivo cada número }
    read( nro );
  end;
  close( arc_logico ); { se cierra el archivo abierto oportunamente con la instrucción rewrite }
end.
```


Archivos → Operaciones adicionales

```
EOF( nombre_logico); (función)
```

- Fin de archivo
- Como trabaja?
- Hay que preguntar primero!!!

```
FileSize(nombre_logico); (función)
```

- Tamaño del archivo

```
FilePos( nombre_logico); (función)
```

- Posición dentro del archivo

```
Seek( nombre_logico, posición); (Procedimiento)
```

- Ir a una posición del archivo
- La posición se cuenta siempre desde el comienzo del archivo
- El primer lugar es el cero .

Archivos → EJ 2 Presentar en pantalla el archivo generado en ej1

```
Procedure Recorrido(var arc_logico: archivo );  
  var nro: integer; { para leer elemento del archivo}  
  begin  
    reset( arc_logico ); {archivo ya creado, para operar debe abrirse como de lect/escr}  
    while not eof( arc_logico) do begin  
      read( arc_logico, nro ); {se obtiene elemento desde archivo }  
      write( nro );           {se presenta cada valor en pantalla}  
    end;  
    close( arc_logico );  
  end;
```

Archivos → Ej 3 Modificación de Datos de un archivo

- Este caso involucra un archivo de datos previamente generado y consiste en cambiar sus datos.
- El archivo debe ser recorrido desde su primer elemento y hasta el último, siguiendo un procesamiento secuencial

{declaración de los tipos de datos necesarios para el problema. Esta declaración se hace efectiva en el programa principal que tiene al proceso Actualizar como uno de sus módulos }

```
Type registro = record  
    Nombre: string[20];  
    Direccion: string[20];  
    Salario: real;  
End;  
Empleados = file of registro;
```

Archivos → Ej 3 Modificación de Datos de un archivo (cont)

```
Procedure actualizar (Var Emp:empleados); {se recibe como parámetro por referencia}
var E: registro;
begin
  Reset( Emp );
  while not eof( Emp ) do begin
    Read( Emp, E);
    E.salario:=E.salario * 1.1;
    Seek( Emp,  filepos(Emp) -1 );
    Write( Emp, E );
  end;
  close( Emp );
end;
```