estudios abiertos
SEAS
GRUPO SANVALERO

A1

A1 UNIDAD DIDACTICA

Estructura de Datos

A. Anexo 1. Ficheros

ÍNDICE

OBJETIVOS

- Conocer el porqué de la existencia de los ficheros, que nos aportan y cuáles son sus peculiaridades.
- Reconocer los distintos tipos de ficheros que existen, y para qué se optimiza su uso.
- Poder aplicar todos los conocimientos adquiridos en aplicaciones que requieran soporte permanente.

INTRODUCCIÓN

El concepto de fichero en C está ligado al manejo de flujos, el propio lenguaje no implementa directamente la gestión de los mismos sino que lo delega en librerías externas. Las principales están estandarizadas por ANSI.

En esta unidad realizaremos un primer acercamiento a la teoría de ficheros para en la unidad siguiente centrarnos en el concepto de flujo y estudiar el manejo de estas entidades en C.

Realmente ya hemos trabajado con flujos, una entrada de teclado es un flujo de entrada mientras que una impresión (por pantalla, por dispositivo de impresión, etc.) es un flujo de salida.

A.1. Qué son los ficheros

Los datos en memoria son volátiles, pero a menudo tendremos la necesidad de que nuestros datos permanezcan, esto es, que sean almacenados en un soporte físico, para satisfacer esta necesidad nos servimos de los ficheros.

DEFINICIÓN

Un fichero es un conjunto de datos el cual está estructurado en una colección de entidades elementales denominadas registros.

Estos registros son por norma general de igual tipo y constan a su vez de diferentes entidades de nivel más bajo llamadas campos.

Si no existiera la posibilidad de almacenar de forma perdurable los datos, estaríamos obligados a tener el software siempre cargado en memoria, con lo que el rendimiento de la máquina caería de forma espectacular si todo el software que pudiera ejecutar estuviera siempre ubicado en la memoria de la máquina.

Además una "caída" de la máquina significaría la pérdida irremediable del software.

Los ficheros se pueden agrupar y relacionar entre sí, existiendo un gran número de programas enfocados a estas tareas.

BÁSICO

Al conjunto de ficheros relacionados se le conoce con el nombre de Base de Datos.

Hasta ahora la metodología de trabajo se puede explicar de forma sencilla bajo el siguiente esquema:

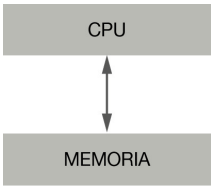

Figura A.1. Esquema de trabajo básico.

Ahora podremos utilizar para nuestros datos generados dinámicamente el siguiente esquema que aporta gran flexibilidad y aprovechamiento del trabajo:

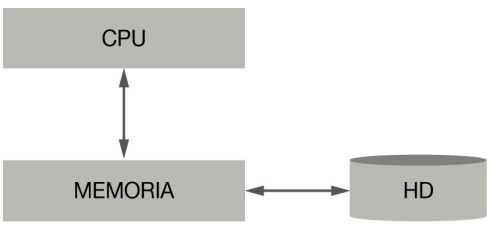

Figura A.2. Esquema de trabajo con la utilización de un HD.

No es posible pasar datos directamente desde la CPU (Unidad Central de Proceso) al HD (Hard Disk o Disco Duro).

Los soportes externos son extremadamente lentos si los comparamos con la velocidad de la CPU, si dicha comunicación fuera directa, para que fuese eficaz dicha comunicación debería seguir el ritmo del dispositivo lento, lo que implicaría un total desaprovechamiento de la velocidad de procesamiento de nuestra máquina.

Para paliar esta deficiencia nos serviremos de lo que se denomina buffer. Un buffer es una memoria intermedia para realizar la comunicación entre la memoria y el disco duro. El esquema que resulta viene a ser:

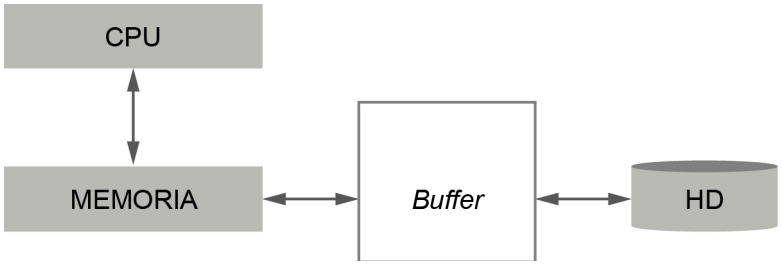

Figura A.3. Esquema de trabajo con la utilización de HD y buffer.

Así el buffer supone un elemento regulador que siempre se interpone entre la memoria y el dispositivo físico de almacenamiento.

El buffer puede localizarse tanto en la controladora del dispositivo, como en la caché de la placa.

Son pocos los lenguajes que permiten la manipulación directa de datos en el buffer, C es uno de esos pocos.

El buffer no es algo prefijado en cuanto a tamaño, sino que podemos redimensionarlo para cada fichero, tipo de acceso, en función de los datos, de la velocidad de obtención de los mismos, etc…

Incluso se le puede conceder a cada fichero su propio buffer específico.

El dimensionar el tamaño del buffer se realiza en tiempo de diseño y es labor del analista de la aplicación.

ATENCIÓN

Es muy importante dimensionar el buffer correctamente pues el rendimiento de la aplicación está estrechamente relacionado con este requisito.

A.2. El concepto de registro

El término registro está ligado al trabajo con buffer.

Al dimensionar el buffer aparecen nuevos conceptos como **Registro Físico** y **Registro Lógico**.

DEFINICIÓN

Registro lógico.
Es la estructura de datos que el usuario se ha definido.

Dicho de otro modo, es su unidad de información.

EJEMPLO

Visualización de un cliente. Impresión en pantalla de todos los datos de un cliente:
INFORMACIÓN: MEMORIA ------> BUFFER

DEFINICIÓN

Registro físico
Es la estructura de datos mínima que se va a transferir en una operación E/S.

EJEMPLO

Lectura del fichero de nóminas de la unidad de disco D:
INFORMACIÓN: DISPOSITIVO ------> BUFFER

Es sencillo vislumbrar una clara relación entre el registro lógico y el registro físico.

A.2.1. Transferencia entre registros físicos y lógicos

Al realizar operaciones de E/S (Entrada / Salida) nos encontraremos en diferentes situaciones en relación a ambas naturalezas de los registros:

Tamaño Registro Lógico Superior al tamaño Registro Físico

ATAJO

Tamaño Registro Lógico > Tamaño Registro Físico

Nuestra unidad de trabajo posee una capacidad mayor a la unidad de transferencia del dispositivo.

EJEMPLO

Tamaño lógico de 100 bytes y físico de 20.

Pongámonos en una situación de operación de lectura de un fichero de texto, tal que:

Estaremos pidiendo 100 caracteres, pero el periférico posee un buffer que sólo le permite devolver 20 de vez, así que para satisfacer nuestra petición se verá obligado a realizar 5 operaciones de lectura (5*20) para conseguir presentarnos esos 100 caracteres.

Obviamente es un sistema lento, pero esta lentitud se agrava si lo implementamos en una red, hasta tal punto que esta podría saturarse hasta caerse, si hubiera muchos accesos concurrentes.

APLICACIÓN

- Este sistema está anticuado y solía estar presente en sistemas operativos monotarea.
- También se usa en operaciones de mantenimiento.

Tamaño Registro Lógico igual al Tamaño Registro Físico

ATAJO

Tamaño Registro Lógico = Tamaño Registro Físico

Este tipo implica que para cada petición se ejecuta una única operación E/S.

BÁSICO

x Peticiones → x Operaciones E/S

Es el caso ideal para operaciones de actualización, aunque parezca que es el caso ideal, no lo es para todo tipo de operaciones.

Operaciones de tipo consulta se pueden mejorar con otro tipo de gestión de buffer como veremos más adelante.

RECUERDA

Siempre que nos referimos a buffer estamos designando una zona específica de memoria.

Normalmente trabajaremos con varios ficheros y la solución óptima será la solución general para todos.

Tamaño Registro Lógico Menor que tamaño Registro Físico

ATAJO

Tamaño Registro Lógico < Tamaño Registro Físico

Nos encontramos en el caso en que pedimos una unidad y el dispositivo nos devuelven una cantidad superior.

EJEMPLO

Por ejemplo pedimos un alumno y el dispositivo nos devuelve 5 alumnos.

Esta gestión presenta varios problemas:

- Mayor consumo de memoria.
- Hasta que no se tienen los x procesados el sistema no graba y si ocurriera algún problema se perderían datos.
- El 'Bloqueo', cuando un usuario pide 1 y se le conceden 5 por defecto estos 5 quedarán bloqueados y no podrá nadie acceder a ellos hasta que queden liberados por quien los solicitó primero.

Existen lenguajes como Cobol que pueden parametrizar este bloqueo (Factor de Bloqueo).

Para realizar consultas este caso es el más óptimo, sin embargo su eficacia es muy pobre para realizar mantenimientos a causa de los registros que permanecen bloqueados y que quizás contengan datos que ni siquiera serán procesados.

A.3. Tipos de ficheros

Los ficheros se pueden clasificar de muchas maneras, en esta unidad estudiaremos una única clasificación en función del acceso a los mismos, pero tenga en cuenta que aunque esta clasificación suele ser la más utilizada debido precisamente al modo de manipulación de los ficheros, existen otras muchas, de tal forma que una misma entidad (fichero) tendrá cabida en diversas clasificaciones.

A.3.1. Ficheros secuenciales

En los ficheros secuenciales un registro se encuentra a continuación de otro, para poder leer el registro n se deberán leer también los n-1 anteriores:

Figura A.4. Diagrama representativo de una estructura de ficheros secuenciales.

La ubicación física de ese tipo de ficheros puede ser cualquier dispositivo.

Operaciones sobre un fichero secuencial

A continuación vamos a ver las diferentes operaciones que podemos definir sobre una estructura de ficheros secuenciales:

- Lectura.
- Añadir registros.

AVISO

Los ficheros secuenciales agregan los registros siempre al final.

- Modificar ficheros existentes.

ERRÓNEO

Tenga en cuenta que no es posible el borrado de registros en ficheros secuenciales.

Los ficheros secuenciales se suelen utilizar para **consultas**, y para la realización de **altas** y **bajas** cuando las tasas son muy pequeñas.

Son muy típicos en la construcción de históricos, donde se apilan datos de acuerdo a un orden cronológico.

Ocupación

La fórmula para conocer el tamaño de un fichero secuencial es la siguiente:

APLICACIÓN

ocupación = longitud_registro x número_registros

Inconvenientes

El mayor inconveniente es la lectura, ya que por para leer el registro 1000 se deben leer antes los 999 anteriores.

No poseen registros claves, ni identificadores, tampoco están ordenados, tan solo apilan datos.

A.3.2. Ficheros de acceso directo

Cada registro ocupa la posición **física** que le corresponde, por tanto dentro del registro deberá existir un campo clave.

AVANZADO

Dicho campo clave almacenará la dirección física donde se almacenan los datos, normalmente el 99 % de las ocasiones se tratará de un campo numérico. Aunque es posible mediante un algoritmo complejo el implementar una clave alfanumérica.

Posteriormente veremos algo más al respecto.

Figura A.5. Diagrama representativo de una estructura de ficheros de acceso directo.

Rn será el registro con la clave n y se encontrará en la posición n.

Dentro de cada Rx habrá un campo que la práctica totalidad de las veces (como ya se ha comentado anteriormente), será numérico e indicativo de la posición donde se almacena dicho registro.

En el momento de asignar la dimensión de dicho campo numérico se reserva ya el espacio para todos los registros.

En los ficheros de tipo directo o de acceso directo se puede acceder a Rn sin pasar por los n-1 registros anteriores.

AVISO

La utilización de estos registros no siempre es la solución ideal, pues en ocasiones se puede infrautilizar el espacio del HD, el factor de rentabilidad que recomienda el uso de los ficheros de acceso directo es de una tasa de ocupación del 75% en adelante.

Se debe destacar que para acceder a los elementos son muy rápidos.

Inconvenientes

El gran inconveniente es que la clave debe ser numérica y para acceder a ellos es totalmente necesaria.

AVANZADO

Ya se ha comentado que existe la posibilidad de imponer una clave alfanumérica, para tal tarea necesitaremos un algoritmo que permita pasar la clave alfanumérica en otra numérica.

Esto sería relativamente sencillo pero la clave debe ser única, pues hace referencia a la ubicación física, que es propia de cada registro.

Este proceso debe ser bidireccional, esto es, se debe cumplir tanto de ida como de vuelta: Dado el campo numérico se debe devolver la clave alfanumérica y viceversa.

El gran problema del algoritmo es que requiere conocer previamente todas las claves de los registros.

Características

Las características principales de una estructura de ficheros de acceso directo las podemos resumir en:

- Claves numéricas pequeñas.
- Pocos Datos.
- Altas tasas de ocupación.

Operaciones sobre un fichero de acceso directo

Veamos cuales se pueden realizar:

- LECTURA (según clave).
- ALTAS (se sitúan en la posición correspondiente previa comprobación de la no existencia del registro).
- MODIFICACIONES (implica existencia previa del registro lo que implica también lectura previa).
- BORRADO (podemos eliminar registros y dejarlos vacíos).

A.3.3. Ficheros indexados

La mayoría de los lenguajes proporcionan el soporte para los 2 primeros tipos de ficheros.

En cambio el soporte para ficheros indexados lo aportan muy pocos lenguajes de programación, el lenguaje COBOL es uno de ellos, puesto que está especializado en el manejo de ficheros; los demás lenguajes tan sólo aportan alguna utilidad.

Ni C ni C++, ni ninguna de sus librerías estándar aportan las funcionalidades necesarias, para el manejo de ficheros indexados. No obstante sí que hay librerías de terceros que las implementan. Otra alternativa es la de crearlas nosotros mismos.

Pero tenga en cuenta que C no fue concebido para este tipo de tareas.

Los ficheros indexados se caracterizan porque conjuntan lo mejor de los ficheros secuenciales y lo mejor de los de acceso directo.

Por tanto podemos afirmar que:

Los ficheros indexados se caracterizarán por un almacenamiento secuencial de los datos (Capacidad Óptima) y un acceso directo a la información (Velocidad).

Este tipo de ficheros permiten el uso de claves alfanuméricas, apareciendo así el nuevo concepto de ‘clave alternativa ’.

ATENCIÓN

Mediante claves alternativas se consigue realizar búsquedas ordenadas según varios campos.

NOTA

La velocidad de un fichero secuencial no es tan alta como la de un fichero de acceso directo, ya que en un indexado siempre hay que recorrer índices, no obstante sus velocidades medias son considerablemente elevadas.

Operatoria de un fichero indexado

Lo primero que contiene un fichero indexado son los datos, esto es, la totalidad de los registros grabados **secuencialmente**, dicho de otro modo, uno tras otro.

Por otro lado necesitamos velocidad de acceso así que existe otro fichero denominado **fichero de índices**, donde se almacenan precisamente la secuencia de índices (también están almacenados en el otro fichero, el de los datos).

Cada uno de los elementos del fichero de índice se corresponde con una dirección física que apunta al fichero de los datos y el campo clave definido por el usuario.

NOTA

El fichero de índices es también de estructura secuencial.

Representación gráfica de un fichero indexado

A continuación se representa un diagrama con la representación gráfica de una estructura de ficheros indexados:

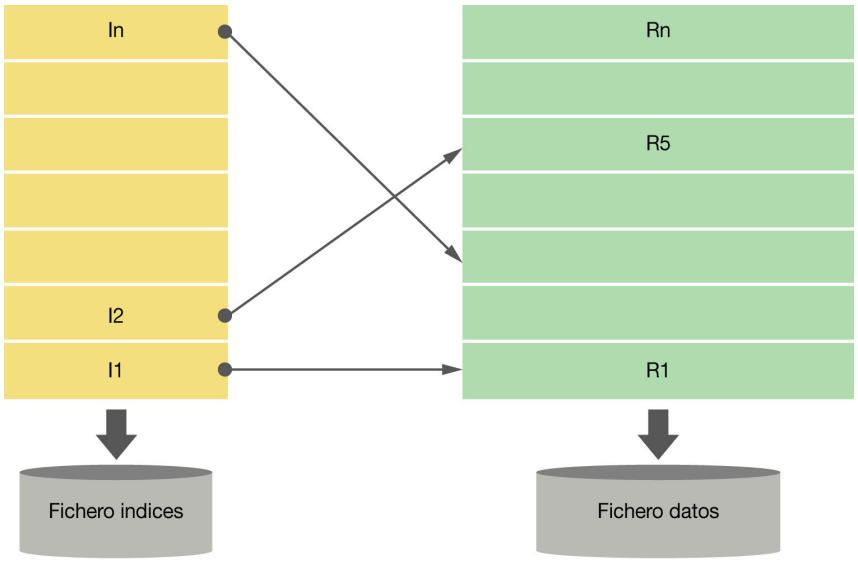

Figura A.6. Diagrama representativo de una estructura de ficheros indexados.

El fichero de índices estará siempre ordenado según la clave bien ascendentemente o bien descendentemente.

Si tenemos varias claves dentro de un fichero de índices habrá un corte o marca donde comenzará otra tabla de índices (en el mismo fichero) ordenada por otros campos y la dirección física de cada uno apuntará igualmente a cada registro del fichero de datos.

Normalmente los ficheros indexados tienen una clave principal que no puede ser duplicada ni modificada, pero las claves alternativas si admitirán modificaciones e incluso duplicados.

Operaciones con fichero indexados

Los ficheros indexados permiten todo tipo de operaciones, dichas operaciones implicarán la manipulación de ambos ficheros y en tantos campos índices como haya.

En el fichero de datos tan solo escribiremos de forma directa, ya que la lectura se realiza indirectamente a través del índice.

El fichero de índices es mucho más pequeño lo que permite un rápido acceso, además se carga en memoria por bloques.

Cuando buscamos en memoria leemos todos los registros secuencialmente hasta llegar al registro x que es el que deseamos (esto es debemos leer también los n-1 anteriores, como en cualquier secuencial) pero una vez encontrado accedemos directamente a la posición física del dato en el dispositivo en el que se encuentra almacenado.

Los registros no se borran simplemente, cuando se quieren eliminar se marcan como no existentes, esto viene heredado de su naturaleza secuencial.

Cada cierto tiempo si hay muchas bajas se suelen reorganizar los índices para recuperar velocidad.

Estas tareas se apoyan en herramientas que suelen venir con el fichero.

RESUMEN

- Un fichero es una colección estructurada de registros de igual tipo éstos a su vez se componen de otras entidades llamadas campos.
- La comunicación con los dispositivos de almacenamiento no es directa sino que se realiza por medio de un buffer.
- Existen registros lógicos, que definimos y manipulamos nosotros y existen los registros físicos propios del dispositivo de almacenamiento.
- La diferencia de tamaño entre ambos genera diversas situaciones que debemos tener en cuenta en el diseño de aplicaciones.
- Según qué operaciones queramos realizar deberemos definir tamaños distintos de registros lógicos.
- La clasificación más común entre ficheros es según su tipo de acceso.
- Los ficheros secuenciales son aquellos cuyos registros de almacenan seguidos y para acceder al registro n debemos recorrer antes los n -1 anteriores.
- Los ficheros de acceso directo permiten acceder directamente al registro, pero requieren espacio inicial para la totalidad de todos sus registros.
- Los ficheros indexados aúnan las bonanzas de los secuenciales y de los de acceso directo.