Clase 3 - Archivos

jueves, 23 de marzo de 2023 15:48

Tipos de archivos

- Sec. de bytes → difícil distinguir inicio y fin
  - ex: de texto.
- Archivos estructurados → registros
  - campos
  - Con long fija o variable
    - + fácil ver dónde arranca y termina cada cosa.
    - ocupa lo justo y necesario, + difícil saber dónde arranca y termina cada cosa.

Campos: unidad + chica de un archivo.

- Identificador de campo:
  - Longitud predefinida (fija) → desperdicio de espacio si sobra, problema si declaro corto y falta.
  - Indicador de longitud: Antes del campo mismo indicador de long. (doble procesamiento)
  - Delimitador al final de cada campo: Caracteres tipo separador (procesamiento + complejo Char a Char)
- Identificador de reg
  - Long. predefinida (cant. de bytes o campos fija)
  - Long variable
    - Indicador de long.
    - Segundo archivo que guarda por ej 1er byte de cada registro.
    - Delimitador: Hay q' leer Char a Char.

Clave → vincula a uno o + campos

Acceder a cada registro específico.

Debe permitir generar orden x el criterio de la llave

primaria → identifica un elemento particular → DNI + Tipo de documento.

secundaria → conjunto de elementos de = valor → Nombre + apellido.

Forma canónica: forma simular (ej: solo mayús, sin espacios)

1. Al ingr. dur nuevo reg.: 1ro forma clave canónica para él
2. Se busca la llave en el arch. si ya existe y el vínculo no se puede ingresar.

Performance

Cant. secuencial
- mejor caso es leer 1 reg, peor leer n.
- promedio: n/2 comparaciones
- orden n → O(n) depende de cant de registros (lineal)

Sí leo de a bloques en lugar de x reg: mejor archivo en disco (menor) = comparaciones

Acceso directo
- Acceso al reg. de una
- Una sola lectura O(1)
- Hay q' saber dónde arranca el reg q' necesito

Nro relativo de reg
- Indica posición relativa respecto al principio del archivo
- Solo funciona con reg. de longitud fija
- Ej: reg 10 y cada reg 100 bytes, voy a byte 1000.

Acceso directo está bueno solo si hay pocos reg. específicos.

si tengo q' acceder a muchos hago muchas cuentas y no va

Formas de acceso

- serie: cada reg se accede después de procesar el anterior, acceso simple
- secuencial: orden de alguna clave
- directo: se accede de una

Cant de cambios

- estáticos
  - pocos cambios
  - se puede actualizar en procedimientos por lotes
  - no necesito estructuras adicionales
- volátiles
  - muchas operaciones
  - organización debe agilizar cambios
  - necesita estructuras adicionales para agilizar

Archivos - operaciones

Baja lógica: Marca de borrado, programa lo ignora pero no lo veo → espacio ocupado al pedo.

Baja física: Borrar borrar (efectivamente)

Con long. fija, agrego y uso fácil. — puedo agregar adelante, atrás o en medio si hubo baja física

Con long variable → puedo querer modificar un reg haciendo q' quede + grande que antes

- soluciones
  - Agregar datos al final del archivo y vincularlo al otro (procesamiento + complejo)
  - Reescribir reg al final (queda espacio vacío en el medio)

Baja lógica

- Hay q' reconocerlos una vez eliminados
- fácil anular elim.
- Programas deben tener cierta lógica para ignorar lo eliminado.

Baja física

- Recupero el espacio — tengo siempre un arch. del tam q' necesito
- Forma + simple es copiar todo a un nuevo archivo pero sin lo borrado
- también puedo compactar (nuevo todo)
- Lo hago cada x tiempo | lo hago cuando necesito espacio.

Aprovechamiento del espacio

- en long. fija es ez
- en long variable hay q' ver q-e entre

para ver dónde meto nuevo reg.

- Búsqueda secuencial: usa marcas de borrado.
  - Busca 1er espacio borrado, sino hay meto al final
  - Lento

Mejor saber de inmediato si hay lugares libres

- si tengo long fija, cant de reg q' necesitan.

↳ Con variable, tengo pos directamente.

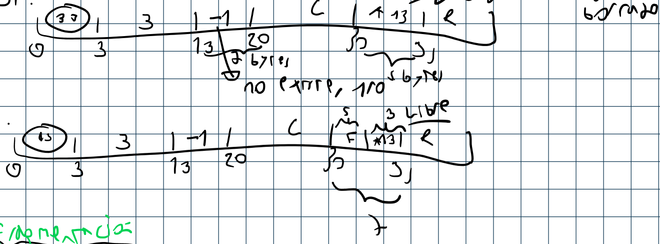
Recuperación de espacio con listas (reg. long fija)

- ↳ Lista encadenada de reg. disponibles
- ↳ Al insertar nuevo reg, cualquier disponible es bueno
- ↳ Lista no necesita orden particular pq todo hueco sirve igual

Reg. con longitud variable

- ↳ Marca de borrado
- ↳ No me sirve cualquier hueco, tiene q' ser >
- ↳ Se usa campo binario q' indica enlace. Conviene q' tenga tamaño.
- ↳ No conviene hacer una pila para guardar pq no me sirve cualquiera
- ↳ Fragmentación. → espacio libre

OF. de dónde arranca el último reg. borrado. → siguiente reg borrado



Fragmentación

- ↳ Interna: desperdicio espacio en un reg.
  - ↳ Generalmente pasa en long fija
  - ↳ long variable evita eso
  - ↳ el espacio asignado no ocupado → string[10]: José (6 bytes al pedo)
- ↳ Externa: Desperdicio espacio entre registros

- Fragmentación externa pasa si hacemos baja logica en reg de long variable

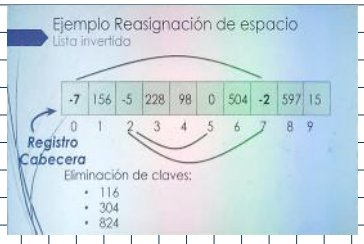
No tienen sentido en longitud fija
Estrategiss de aprovechamiento de espacio en reg de long variable
-> mejor ajuste: busca el espacio cuyo tamaño se acerque más al de lo que quiero insertar. Exige búsqueda. Genera fragmentación interna.
-> primer ajuste: primero que encuentra (dónde entre). Menos busqueda. Asigna el espacio completo. Frangemntacion interna, me queda espacio reservado al pedo
-> peor ajuste: elije la entrada más grande, se le asigna solo el espacio necesario y el resto queda libre para otro reg. Fragmentación externa. Tiene que recorrer todo el coso buscando el más grande
En mejor y primer no siempre recorro toda la lista, en peor sí.

Bajas físicas → Puedo copiar todo el arch sacando lo borrado
↳ Puedo poner el último dato en donde está lo q' quiero borrar (sirve solo si no está ordenado)

Baja lógica → Recorrer desde el principio hasta el 1er reg borrado y meter ahí el nuevo. → Reasignación de espacio
↳ Recuperación de espacio → Cada tanto borro físicamente todo.

Lista invertida (pila) → Dentro del archivo.
↳ Contiene posiciones disponibles en el archivo.

Registro cabecera → en el primer reg. del arch tengo apuntado al 1er elem borrado
Guardo en cabecera pos de 1er borrado y en pos de 1er borrado lo q' había en cabecera
→ Se guarda -5 para distinguirlo de un dato válido


Ejemplo Reasignación de espacio
Lista invertida
-7 156 -5 228 98 0 504 -2 597 15
0 1 2 3 4 5 6 7 8 9
Registro Cabecera
Eliminación de claves:
• 116
• 304
• 824