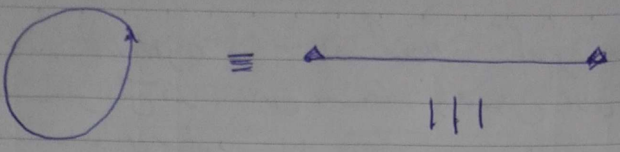
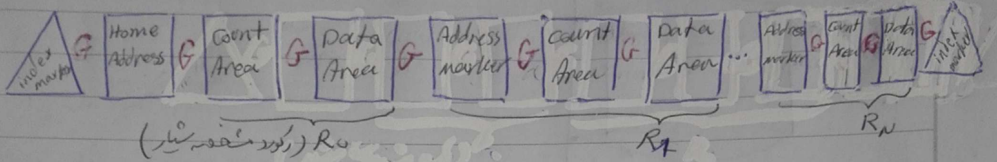


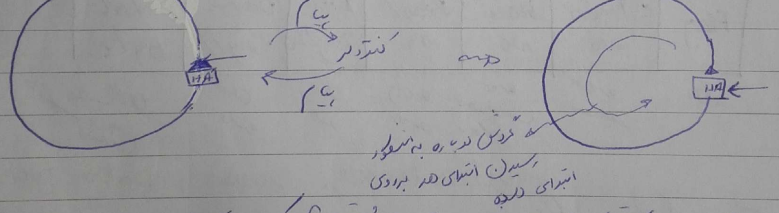

index marker
G
Home Address
Count Area
Data Area
Address marker
Count Area
Data Area
...
Address marker
Count Area
Data Area
index marker
(رکورد مشخصه شیار) R0
R1
RN

① index marker: هر علامت خاصی است بر روی شیار که مشخص کننده نقطه شروع فیزیکی شیار است.

Gap ها در دیسک: با توجه رفت و برگشت پیام ها در پردازش کنترلر روی آن برای جلوگیری از گردش اضافی دیسک، Gap ها در زمان نوشتن ایجاد می‌شوند.
نه زمان خواندن

توجه: برخلاف نوار، طول Gap ها در Disk ها متفاوت می‌باشد.

توجه: اگر Gap ها در Disk وجود نداشته باشد (اطلاعات به یکدیگر چسبیده باشند) مثلاً هنگام چک کردن Home Address و رفتن و برگشت پیام هد به میانه داده می‌رسد و دوباره باید بگردد و ابتدای آن به زیر هد برود.

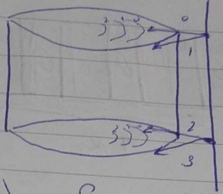

HA
پیام
کنترلر
پیام
هد
HA
گردش دوباره به منظور رسیدن ابتدای هر رکوردی ابتدای داده

② Home Address: عبارت است از مشخصات شیار که عبارت است از چهار فیلد اطلاعاتی:

۶ بایت	2 بایت	2 بایت	2 بایت
Flag	cylinder No	Head No	control Byte

Flag: شامل اطلاعاتی درباره وضعیت شیار می‌باشد و نشان می‌دهد که آیا شیار قابل استفاده است یا فعلاً خراب می‌باشد و در صورت خراب بودن شیار، از چه شیار جایگزینی استفاده می‌شود.

Cylinder No و Head No: آدرس شیار مخصوص را روی دیسک مشخص می کنند. این مشخصه به خصوص برای آدرس دهی مستقیم استفاده می شود.



Control Byte: برای ردیابی error‌ها شبیه (مثل parity bit) که در قبل گفته شد (مربوط به سیستم عامل)

④ ③
Count area - Data area

R0 (رکورد شماره صفر - رکورد مشخصات شیار): این رکورد حاوی اطلاعاتی راجع به چگونگی سازماندهی داده در ادامه شیار می باشد. این رکورد مشخصات شیار خود به دو قسمت Count Area و Data Area تقسیم می شود. در بخش Count Area رکورد مشخصات شیار، هفت عنصر اطلاعاتی (هفت Field) که جمعاً 11 بایت را اشغال می کنند به شرح و شکل زیر وجود دارند:

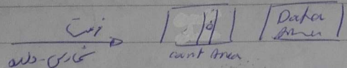
1	2	2	1	1	2 بایت	2 بایت
Flag	Cylinder No	Head No	Record No	Key length	Data Length	Control Byte
فلگ	شماره سیلندر	شماره هد خواندن و نوشتن	شماره رکورد	طول کلید	طول داده	بایت های کنترل

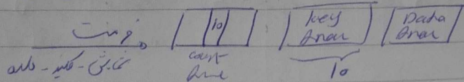
Data area: اطلاعات در Data area ذخیره می گردد.

فیلد Flag و فیلدهای Cylinder No و Head No هم نقش قبل در Home Address را دارد که در اینجا تکرار شده اند طوری که اگر داده ها از شیاری به شیار جانشین انتقال پیدا کنند این قسمت تمام اطلاعات لازم را در مورد بقیه اطلاعات به سیستم می دهد.

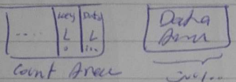
شماره رکورد: شماره ردیف رکورد را روی شیار نشان می دهد

طول کلید: برای زمانی استفاده می‌شود که از فرمت نوع دوم یعنی فرمت شمارش - کلید - داده استفاده می‌کنیم.


فرمت
شمارش - داده
0
count Area
Data Area


فرمت
شمارش - کلید - داده
10
count Area
key Area
10
Data Area

طول داده: طول داده هایی که جدا اند آن ورد می‌گیرد یا بخصوص زمانی که طول داده ها متغیر اند ذخیره می‌کنند.


....
key
L
0
Data
L
100
Count Area
Data Area
۱۰۰ بایت

بایت های کنترلی: برای کنترل خطاهایی است که در قسمت Data Area رخ می‌دهد.

⑤ Address marker: یک فیلد دو بایتی است که معرف شروع رکورد می‌باشد و مورد استفاده واحد کنترل یا کنترلر هد خواندن و نوشتن می‌باشد.

فصل ۲

«تعاریف مبنایی»

فیلد (Field): کوچکترین واحد اطلاعاتی که اگر شکسته شود معنی و مفهوم خود را از دست می‌دهد

رکورد (Record): مجموعه‌ای از فیلدها که به صورت منطقی به یکدیگر وابسته هستند را گویند.

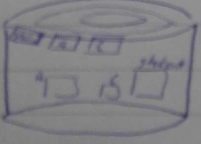
رکورد از نقطه نظر ساختار (منطقی)
- رکورد با قالب ثابت و یکسان: طول همه رکوردها در همه فرم‌ها ثابت و مکان فیلدها همواره ثابت
- رکورد با قالب غیرثابت و غیریکسان

مثال برای قالب ثابت و یکسان:

ID	Name	L-Name
100	علی	
200	محمد علی	
300	محمد علی حسن	
400		

ID C(3)
Name C(10)
LName C(12)
طول رکورد ← 25

تعریف File Directory: فهرستی از آدرس‌ها و غیره که در دو سیلندر اول ذخیره می‌شود. در حقیقت برای جستجوی اطلاعات لازم نیست کل دیسک گشته شود، کافی است سیستم File Dir را برای جستجوی آن فایل سرچ کنیم



× 100

B + G ⸱⸱⸱ 80 ÷ 800

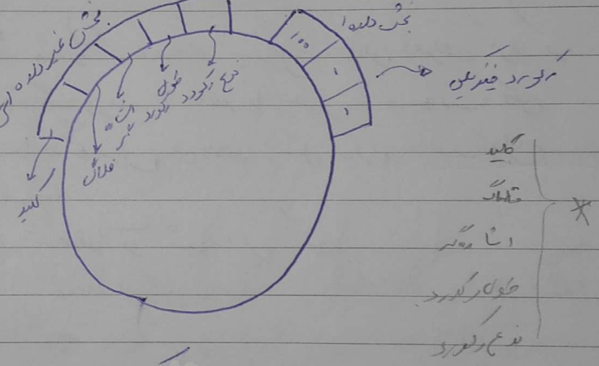
1600 × 0/5

رکورد از نقطه نظر ذخیره سازی (فیزیکی):

ID	Name	Lname
100	—	—
200	—	—
300	—	—
400	—	—

رکورد منطقی ها

دید منطقی


بخش داده ها
بخش غیر داده ای
100
1
1
رکورد فیزیکی
نوع رکورد
طول رکورد
اشاره گر
فلاگ
کلید
کلید
فلاگ
اشاره گر
طول رکورد
نوع رکورد
*

دید فیزیکی

نوع رکورد:

اگر فایل ما مانند فایل دانشجویی بود (فایل تک نوعی) دیگر نیازی به نوع رکورد نداریم. ولی اگر فایل ما از نوع فایل چند نوعی بود مانند فایل انبار که در آن مثلاً یک فایل رسید داریم یک فایل حواله، آنگاه نوع رکورد برای رسید مثلاً 1 و نوع رکورد برای حواله 2 قرار می دهیم تا از هم قابل تشخیص باشند

طول رکورد: { ثابت (حالت ثابت و مکانی)
متغیر (حالت غیر ثابت و غیر مکانی)

برای رکورد ها با طول متغیر ابتدا آن فیلدی به نام فیلد طول رکورد قرار می گیرد که به وسیله آن بتوان رکورد را تشخیص داد.

pointer :

اشاره گر : (عملاً اشاره گر از pointer : دید منطقی را در محیط فیزیکی پیاده سازی می کند)

فیلدی است که مقدار آن آدرس رکورد بعد از آن را در بر می گیرد. زیرا در بسیاری از موارد رکوردها به صورت منظم ممکن است پشت هم ذخیره نشوند ولی برای این که بتوان دید منطقی به این محیط فیزیکی داشت باید آدرسی وجود داشته باشد تا رکورد قبلی را به رکورد بعدی متصل کند

★ اشاره‌گر از رکورد به رکورد، رکورد به بلاک، بلاک به رکورد، بلاک به بلاک، رکورد به فایل اشاره می کند ★

فلاگ : مقداری است که درون آن اطلاعات وضعیتی یا راهنما رکورد را مشخص می کند

انواع فلاگ ها:
- عملیاتی
  - ① برای کاری که قرار است در آینده روی رکورد انجام شود (مثل حذف منطقی و حذف فیزیکی)
  - ② برای کاری که روی رکورد انجام شده (مثلاً update برای flag)
- حفاظتی : مثلاً به عنوان Admin می توان فلاگی تعریف کرد که کاربران بتوانند یک رکورد را ببینند یا خیر. در حالت کلی اجازه این کار را به کاربران می دهد (مثل دیدن، حذف کردن و ...)

کلید (شناسه) : ترکیبی از چند فیلد برای شناسایی رکورد

① کلید منتخب :

② کلید اصلی : شناسه‌ای از رکورد که چند خاصیت دارد
- ① منحصر به فرد است
- ② کمترین طول را دارد
- ③ نیاز کاربر را برطرف خواهد کرد

③ کلید جانشین : هر کلید غیر از کلید اصلی را کلید جانشین گویند

④ کلید خارجی : ارتباط بین جداول و فایل ها را ایجاد می کند.

⑤ ابر کلید : همان کلید منتخب است ولی می تواند خاصیت حداقل بودن را نداشته باشد.

فایل : مجموعه رکورد ها که منطقاً به یکدیگر وابسته هستند

فایل به صورت کلی به 2 دسته تقسیم می‌شوند.

① فایل‌های داده‌ای
- فایل‌های دودویی (Binary) : فایل‌هایی که قابل ویرایش با نرم افزارهای معمولی نیستند.
- فایل‌ها ASCII : فایل‌هایی که به برخی محیط‌ها نرم افزار قابل ویرایش هستند (مانند Word)

② فایل‌های سیستمی : فایل‌هایی که خود سیستم ایجاد می‌کند و گاهی به کاربر اجازه استفاده از آن‌ها را می‌دهد (مانند File Directory که یک فایل سیستمی است)

یا File Allocation table

FAT

مشخصات تمام فایل‌های داده‌ای روی Disk را در خودش دارد. دارای دو نسخه است که نسخه دوم (یا سطر دوم) به عنوان پشتیبان ذخیره می‌شود.

بلاک (Block) : واحد تبادل اطلاعات بین حافظه اصلی و حافظه جانبی است و داخل آن می‌توان 1 تا چند رکورد قرار گیرد. (B)

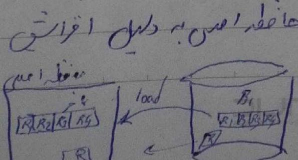
فاکتور بلاک : $B_f = \left\lfloor \frac{B}{R} \right\rfloor$

مزایا بلاک بندی : ① با انجام بلاک بندی باعث کاهش Gap ها می‌شویم در نتیجه فضای هرز کمتری ایجاد می‌شود.

② با کاهش فضای هرز، طول فضای فایل کاهش یافته و در نتیجه سرعت دستیابی افزایش می‌یابد (سرعت بیشتر پردازش)

③ با بلاک بندی رکوردها، تعداد عملیات I/O کاهش می‌یابد (سرعت بیشتر پردازش)

معایب بلاک بندی : ① افزایش مصرف حافظه اصلی به دلیل افزایش طول بافرها



② عمل Blocking و DeBlocking نیاز به نرم افزار هست

③ احتمال بروز اشتباه به دلیل حجم بالای اطلاعات منتقل شده

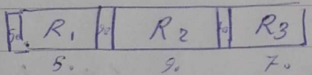
روش های تعیین محدوده رکورد در داخل یک Block

اگر رکورد با طول ثابت باشد : در داخل FAT طول هر رکورد ذکر می شود (مثلاً R=20) پس محدوده رکورد در داخل Block مشخص می شود.

اگر رکورد با طول متغیر باشد : برخلاف نوع قبلی در FAT طول ذکر نمی شود

پس از روش های زیر استفاده می شود

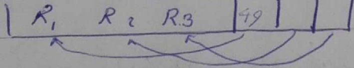
① قرار دادن فیلد طول در ابتدای هر رکورد



② قرار دادن علامتی خاص در انتهای هر رکورد

| R1 * R2 * R3 * |

③ استفاده از جدول مکان ها



یعنی یک رکورد فقط می‌تواند در یک بلاک جا بگیرد و نمی‌تواند
و بین دو بلاک تقسیم شود

تکنیک های بلاک بندی

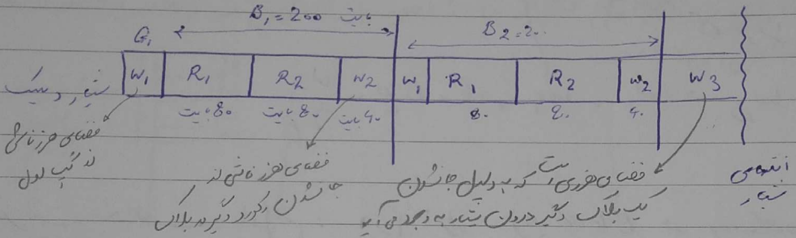
1- تکنیک بلاک بندی رکورد ها با طول ثابت و یکپارچه — unspanned

2- " " " " " " با طول متغیر و دوپارچه — spanned

3- " " " " " " با طول متغیر و یکپارچه — unspanned

با طول ثابت و یکپارچه:

مثال:

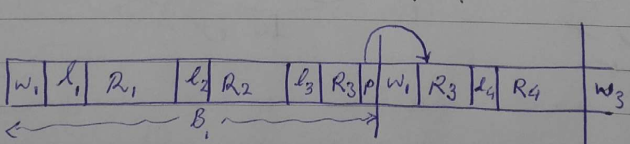

G_1
$B_1 = 200$ بایت
$B_2 = 200$
W_1
R_1
R_2
W_2
W_1
R_1
R_2
W_2
W_3
80 بایت
80 بایت
40 بایت
80
80
40
شیار در دیسک
فضای هرز ناشی از گپ اول
فضای هرز ناشی از جا نشدن رکورد دیگر در بلاک
فضای هرزی است که به دلیل جا نشدن یک بلاک دیگر درون شیار به وجود می‌آید
انتهای شیار

فاکتور بلاک: $B_f = \left\lfloor \frac{B}{R} \right\rfloor = \left\lfloor \frac{200}{80} \right\rfloor = 2$

مقدار حافظه هدر رفته در هر بلاک: $W_B = W_1 + W_2 + \frac{W_3}{T_f}$

T_f ← تعداد بلاک های موجود در شیار

با طول متغیر و دوپارچه:


W_1
l_1
R_1
l_2
R_2
l_3
R_3
P
W_1
R_3
l_4
R_4
W_3
B_1

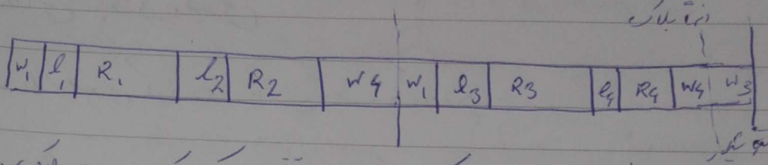
$$B_f = \frac{B-P}{R+l}$$

R ← متوسط طول رکورد

$$W_B = W_1 + P + B_f \times l + \frac{W_3}{T_f}$$

T_f ← تعداد بلاک ها در شیار

با طول متغیر یکپارچه


انتهای بلاک
W_1
l_1
R_1
l_2
R_2
W_4
W_1
l_3
R_3
l_4
R_4
W_4
W_3
انتهای رکورد

W_4 : حافظه هرزی است که به دلیل عدم تقسیم یک رکورد به دو بلاک بوجود آمده و طول آن متغیر است.

طول بلاک

متوسط

$$Bf = \frac{B - W_4}{R + l}$$

متوسط طول رکورد

$$W_B = W_1 + W_4 + l \times Bf + \frac{W_3}{Tf}$$

مزیت حالت اول: نرم افزار Blocking و DeBlocking نسبتاً تکنیک ساده تری به مقایسه با دوم و سوم دارد

اشکال حالت اول: انعطاف بسیار کمتر نسبت به حالت دوم و سوم

مزیت حالت دوم: فضای هرز کمتر از دو حالت دیگر دارد.

اشکال حالت دوم: اگر رکوردی بین دو بلاک تقسیم شود برای بازیابی آن دو بار عمل خواندن انجام می شود و سرعت خواندن نسبت به تکنیک سوم کمتر است

دو بار عمل IO

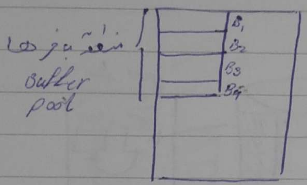
بافر

بافر نوعی حافظه‌ی واسطه می‌باشد، برای هماهنگ سازی سرعت بین دستگاه‌های ورودی و خروجی و CPU.

انواع بافر

1. سخت افزاری: در سخت افزار دستگاه موجود است.

2. نرم افزاری: در اصل بخشی از حافظه اصلی را از طریق یک برنامه به تعدادی از بافر ها اختصاص می دهد.



نحوه ایجاد بافرها:

1. وقتی فایلی باز می شود سیستم عامل بصورت خودکار، تعدادی بافر در اختیار فایل قرار می دهد.
(معمولاً یک بافر ورودی و یک بافر خروجی در اختیار فایل قرار می دهد)

2. خود کاربر از سیستم درخواست بافرهایی می کند

Config.sys → Buffers = 70

توجه: همیشه تعداد زیاد بافرها باعث سرعت بخشیدن به دستگاه نیست بلکه گاهی زیاد بودن بافر باعث اشغال فضای حافظه در نتیجه نداشتن حافظه مناسب برای اجرای برنامه ها و در نتیجه کند شدن سیستم است.

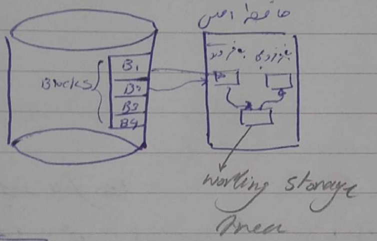
چگونگی دستیابی سیستم (CPU) به محتوای بافرها:

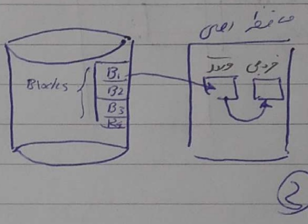
۱- روش انتقالی (Move mode)

در این روش اطلاعات بافر وارد بافر ورودی به حافظه اصلی شود، بلافاصله وارد Working storage Area شده و CPU آن را پردازش می کند، در این روش به علت وجود Working storage می توان همزمان با پردازش CPU بر روی Working storage بافر دوم را وارد حافظه اصلی کرد

2- روش مکان نمایی (Locate mode)

برخلاف روش اول حافظه سومی به نام Working storage Area وجود ندارد.

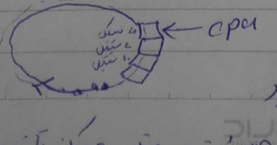

حافظه اصلی
بافر ورودی
بافر دوم
Blocks
B1
B2
B3
B4
Working storage Area
۱


حافظه اصلی
خروجی
ورودی
Blocks
B1
B2
B3
B4
2

توجه: در روش اول سرعت پردازش بالاتر می رود ولی فضای بیشتری از حافظه اصلی گرفته می شود (ولی در روش دوم سرعت پایین و حافظه اشغالی کمتر است)

مدیریت بافرها: (۱- شبکه ۲- چندکاربر)

بسته به نوع عملیات و نوع اطلاعات درون بافرها، پردازش بافرها ممکن است مقادیر متفاوتی طول بکشد، اصطلاحاً به آن سیکل گفته می شود. بدین معنا که CPU در زمان مشخصی (مثلاً چند میلی ثانیه) ۱ سیکل از بافر اول را پردازش کرده و به سراغ بافر بعدی می رود، در این حالت لیستی مدور از بافرها ایجاد می شود که مدام توسط CPU پردازش می شوند و وقتی تعداد کل سیکل های آن تمام شد بافر خالی شده و عملیات جدید جایگزین می شود


۴ سیکل
۲ سیکل
۱۰ سیکل
CPU

در برخی مواقع به بعضی عملیات اولویت داده می شود
بدین معنی که CPU برای آن عمل خاص زمان بیشتری صرف می کند تا زودتر تمام شود

مثال: فرض کنید 30 کاربر که ۱ واحد در حال کار کردن با برنامه حسابان باشند اگر هر کاربر که سه فایل و هر فایل که دو بافر استفاده کند و حجم حافظه اصلی برابر 1MB و اندازه بافرها 1KB باشد، مشخص نمائید چند درصد از حافظه صرف بافرینگ می‌شود؟

30 × 3 = 90 فایل

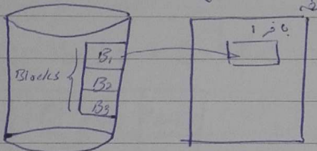
90 × 2 = 180 تعداد بافرها

180 × 1KB = 180 KB

$\frac{180\ KB}{1000\ KB} \times 100 = \%18$

بافرینگ:

۱- بافرینگ ساده (Singel Buffer): در این نوع بافرینگ از یک بافر استفاده می‌شود



وقتی بافر خالی است cpu بیکار

زمان پردازش cpu دستگاه‌های ورود خروجی بیکار

در این حالت چون سرعت cpu خیلی بالاتر از دستگاه‌ها ورود خروجی است cpu خیلی بیشتر بیکار خواهد ماند.

2 بافرینگ ~~مضاعف~~ مضاعف (Double Buffer): در این نوع بافرینگ از دو بافر استفاده می‌شود، در این حالت نسبت به حالت قبلی cpu کمتر بیکار خواهد ماند و cpu و دستگاه‌های ورودی خروجی می‌توانند همزمان کار کنند.

3- بافرینگ چندگانه (multi Buffer)

شرایط استفاده از بافر چندگانه:
- ۱- پردازش فایل ترتیبی باشد *
- ۲- امکانات حافظه وجود داشته باشد (حافظه اصلی به مقدار زیاد)

توجه: محتویات بافرها هیچگاه بعد از پردازش بلافاصله پاک نمی شوند، زیرا ممکن است رکوردی را دوباره لازم داشته باشیم. در حقیقت بافرها زمانی خالی می شوند که بعد از گشتن بافرها و مطمئن شدن نبود بافر، یک بافر خالی می شود.

مثال: فایلی داریم با 10000 رکورد 80 بایتی که آن را روی نوار با چگالی 1600bpi و طول Gap های 0,5" قرار داده ایم. اگر فاکتور بلاک را 60 بگیریم چند درصد از این نوار به صورت مفید استفاده خواهد شد؟

$\boxed{B \mid G}$

تعداد رکورد = 10000

$R = 80$ بایت

(چگالی) $D = 1600bpi$

$B_f = 60 \longrightarrow \left\lfloor \frac{B}{\underset{80}{R}} \right\rfloor = 60 \longrightarrow B = 4800$ بایت

$G = 0,5 \times 1600 = 800$ بایت

$$\frac{B}{B+G} \times 100 = \frac{4800}{4800+800} \times 100 = \%85,71$$

چگالی لود اولیه

زمانی که می خواهیم فایل را روی دیسک ذخیره کنیم، اگر بدانیم که بعداً بیشتر قرار است روی آن فایل اطلاعات بیشتری ذخیره شود، زمان ایجاد آن فایل، فضای بیشتری به آن فایل اختصاص می دهند.

مزایای چگالی لود اولیه: اگر چنانچه بتوان برای فایلی که امروز ایجاد می کنیم مشخص کرد که در آینده به چه میزان به حجم آن اضافه خواهد شد می توان فضایی را در همان ابتدا برای این فایل اختصاص داد، با این کار از مزیت زیر برخوردار می شویم:

① ناحیه رزرو [illegible] می شود تا از پراکندگی شکست رکوردها

7 آذر میان ترم

سعی رسانه فشرده شود، به عبارتی locality یا همسایگی رکوردها در فایل بهتر حفظ شود

② ناحیه رزرو باعث می شود تا عملیات روی فایل مثل درج یک رکورد یا حذف به شکل ساده تر صورت گیرد

معایب جای خالی بود: در ابتدا حافظه هدر دادیم.

توجه: $\frac{l}{B} < 1$

l طول اطلاعات

B طول بلاک

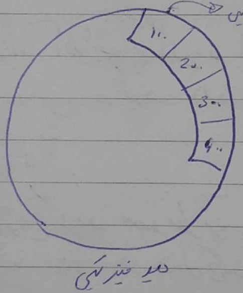
locality (همسایگی)

میزان همسایگی رکورد های منطقاً همجوار در محیط فیزیکی

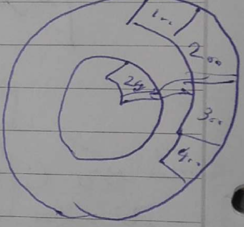
ID	name	lname
100		
250		
300		
400		

رکورد منطقی

دید منطقی



دید فیزیکی
(الف)



(ب)

* در حالت (الف) دید فیزیکی بسیار به دید منطقی نزدیک است و locality بسیار قوی است

در حالت (ب) رکوردی اضافه شده است و چون فضایی برای آن پیش بینی نشده است، رکورد ذخیره شده با پوینتر به دید منطقی نزدیک می شود

در این حالت locality ضعیف تر از حالت قبلی است

سازماندهی مجدد فایل: برای دوباره قوی کردن locality و افزایش سرعت

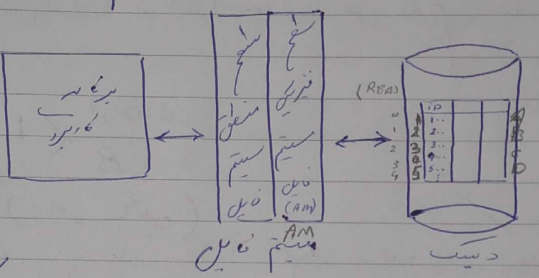
یعنی فایل را مجدداً می خوانیم، رکوردهای که حذف منطقی شده اند حذف فیزیکی شده، و رکوردهایی که در جای درست از نظر منطقی قرار نگرفتند در جای اصلی قرار می گیرند و locality دوباره به حالت قوی بر می گردد.

نشانی دهی در سیستم فایل

فرض کنید یک برنامه کاربردی می‌خواهد یک فایل که قبلاً ذخیره شده را بازیابی و از آن استفاده کند.

برنامه به خودی خود نمی‌تواند این کار را انجام دهد و سیستم عامل واسط می‌شود.

سیستم عامل به وسیله (سیستم فایل) که یک برنامه واسط است این کار را انجام می‌دهد


برنامه کاربرد
سطح منطقی سیستم فایل
سطح فیزیکی سیستم فایل (AM)
AM
سیستم فایل
(RBA)
0
1
2
3
4
ID
1..
2..
3..
4..
5..
1
2
3
4
5
A
B
C
D
دیسک

توجه: برای بازیابی اطلاعات نیاز به آدرس رکورد مورد نظر داریم که چند شکل زیر برای آدرس دهی وجود دارد:

آدرس در سطح برنامه کاربرد سیستم فایل:

① آدرس محتوایی: به دنبال محتوای رکورد مورد نظر برای بازیابی آن می‌گردد مثلاً:
رکوردی با ID ۲۰۰ را بازیابی می‌کند ـــــ ID = 200

② آدرس نسبی: به دنبال شماره رکورد برای بازیابی آن می‌گردد مثلاً
رکوردی با شماره رکورد ۱ را بازیابی می‌کند ـــــ RecNo = 2

③ آدرس مجازی: به دنبال نامی که برای رکورد ذخیره کردیم می‌گردد، مثلاً
رکوردی با نام مجازی B را بازیابی می‌کند. ـــــ B

آدرس در سطح منطقی :

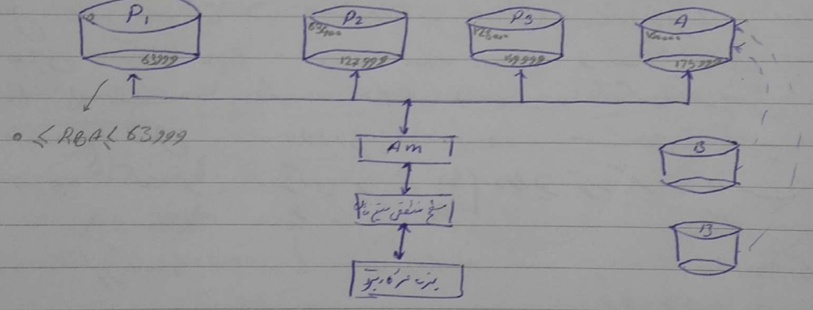
هر سه مدل آدرس دهی قبلی در عمل به یک آدرس واحد به نام آدرس نسبی بلاک (Relative Block Address) تبدیل می شود.

هر سه مثال قبلی RBA = 1 را نشان می دهد

آدرس فیزیکی :

این آدرس (RBA) در انتها توسط یک الگوریتم به آدرس فیزیکی (شماره درایو، شماره سیلندر ، شماره شیار ، شماره دیسک) تبدیل می شود

آدرس فیزیکی $\xleftarrow{\text{الگوریتم}}$ آدرس سطح منطقی ⟷ آدرس سطح برنامه کاربرد


P1
63999
P2
127999
P3
A
175999
B
B
0 ≤ RBA ≤ 63999
Am
سطح منطقی سیستم
برنامه کاربر

مشخصات دیسک ها :

نوع درایو	تعداد	تعداد سیلندرها	تعداد شیارها در سیلندر	تعداد بلاک ها در شیار	کل تعداد بلاک تعداد بلاک در یک دیسک
دیسک های ثابت	2,5	400	20	8	$8 \times 20 \times 400 = 6400$ $6400 \times 2,5 = 160,000$ (ظرفیت سه دیسک ثابت)
دیسک های جابه جا شدنی	3	200	20	4	$4 \times 20 \times 200 = 4800$ $4800 \times 3 = \underline{208006}$ ظرفیت سه دیسک متحرک

چگونگی پیدا کردن آدرس درست فیزیکی سیستم فایل

نوع درایو	شروع RBA	پایانی RBA	آدرس فیزیکی
P_1	0	63999	$\text{شماره سیلندر} = \left\lfloor (\overset{\text{مورد نظر}}{RBA} - \overset{\text{شروع}}{RBA}) / (20 \times 8) \right\rfloor$
P_2	64000	127999	
P_3	128000	159999	$\text{شماره شیار} = \left\lfloor (\overset{\text{مورد نظر}}{RBA} - \overset{\text{شروع}}{RBA}) \bmod (20 \times 8) / 8 \right\rfloor$

تعداد شیار در سیلندر — تعداد بلاک در شیار

$$\text{شماره بلاک} = (\overset{\text{مورد نظر}}{RBA} - \overset{\text{شروع}}{RBA}) \bmod 8$$

A_2	160000	175999	$\text{شماره سیلندر} = \left\lfloor (\overset{\text{مورد نظر}}{RBA} - \overset{\text{شروع}}{RBA}) / (20 \times 4) \right\rfloor$

$$\text{شماره شیار} = \left\lfloor (RBA - RBA) \bmod (20 \times 4) / 4 \right\rfloor$$

$$\text{شماره بلاک} = (RBA - RBA) \bmod 4$$

مثال: یک دیسک ثابت با مشخصات زیر داریم. تعداد سیلندر های دیسک 300، تعداد شیار ها 15 و تعداد بلاک ها در شیار 10 است. اگر RBA بدست آمده از بخش منطقی سیستم فایل 59999 باشد، مشخص نمایید این بلاک در چه آدرس فیزیکی و در کدام درایو قرار گرفته است

تعداد شیار: 15

تعداد بلاک: 10

RBA مورد نظر: 59999

$300 \times 15 \times 10 = 45000$

دیسک اول: 0 تا 44999

دیسک دوم: 45000 تا 89999 ✓✓

$45000 \times 2 = 90000$

$$\text{شماره سیلندر} = \left\lfloor \overbrace{(59999 - 45000)}^{19999} / (15 \times 10) \right\rfloor = 99$$

$$\text{شماره شیار} = \left\lfloor (59999 - 45000) \bmod (15 \times 10) / 10 \right\rfloor = 14$$

$$\text{شماره بلاک} = (5999 - 45000) \bmod 10$$

$$14999 \div 150 = \boxed{99} \quad \text{شماره سیلندر هست}$$

$$14999 - 14850 = 149$$

$$149 \div 10 = \boxed{14} \quad \text{شماره شیار است}$$

$$\boxed{9} \quad \text{شماره بلاک}$$

ارزیابی کارآیی رسانه‌ها ذخیره‌سازی (دیسک و نوار)

نوار:

مفروضات

B طول بلاک

G " گپ

d چگالی نوار

l طول نوار

t نرخ اسمی انتقال و نرخ واقعی انتقال t'

$btt = \frac{B}{t}$ زمان خواندن یک بلاک (msec)

$t = \frac{B}{btt}$

$\frac{G}{t}$ زمان خواندن یک گپ (ms ثانیه)

A زمان ایستادن و حرکت نوار (msec)

n تعداد بلاک‌ها

ظرفیت اسمی = $l \times d$

ظرفیت واقعی = ظرفیت اسمی × نسبت اشغال واقعی

$$= l \times d \times \frac{B}{B+G}$$

$$t' \text{(نرخ واقعی انتقال)} = \frac{\overset{\text{تبدیل به ثانیه}}{\frac{btt}{1000}}}{\underbrace{\frac{btt}{1000} + \frac{G}{t} + \frac{A}{1000}}_{\text{زمان واقعی خواندن یک بلاک}}} \times t$$

هر بار فقط یک بلاک خوانده شده سپس نوار می‌ایستد (الگوی بلاکی)

$$t' \text{(نرخ واقعی انتقال)} = \frac{n\frac{btt}{1000}}{\underbrace{n\frac{btt}{1000} + n\frac{G}{t} + \frac{A}{1000}}_{\text{زمان واقعی خواندن چند بلاک}}} \times t$$

وقتی ما n بار چند بلاک خوانده شده و سپس نوار می‌ایستد (الگوی جویبار)

تذکر: نرخ واقعی انتقال در نوار بستگی به عواملی مثل طول بلاک، طول گپ، نرخ اسمی انتقال خواهند داشت.

دیسک:

ظرفیت اسمی یک دیسک $= S$

ظرفیت واقعی " " $= S - \sum_{i=1}^{n} w_i$ حافظه هرز

w_i:
- 1- تکنیک‌های بلاک‌بندی
- 2- طول Gap
- 3- فرمت بندی
- 4- طول بلاک
- 5- هرگاه نسبت تعداد درستی رکوردها را به تعداد درستی سکتورها بالا (فرمت بندی سخت افزاری)

مثال: فرض نمائید فایلی که طول هر رکورد ۱۶۰ بایت می باشد. این فایل را روی دیسکی که به شکل سخت افزاری فرمت شده و طول سکتورهای آن 256 است. مشخص نمائید اگر فاکتور بلاک را از ۱ تا 8 تغییر دهیم چند درصد از آن شیار به صورت موثر استفاده خواهد شد.

درصد استفاده واقعی از شیار	ظرفیت اشغالی	تعداد سکتور هر بلاک	ظرفیت واقعی هر بلاک	Bf
$\frac{160}{256}\times 100 \simeq \%63$	256	1	160	1
$\frac{320}{512}\times 100 \simeq \%63$	512	2	320	2
$\frac{480}{512}\times 100 \simeq \%93$	512	2	480	3
				4
$\frac{1280}{1280}\times 100 = \%100$	1280	5	1280	8

پس :

$$\text{درصد استفاده واقعی از شیار} = \frac{\overset{\text{طول رکورد}}{R} \times \overset{\text{تعداد رکورد}}{Bf}}{\underset{\text{که طول سکتور}}{S} \times \underset{\text{تعداد سکتور در هر شیار}}{N}} \times 100$$

مثال: اگر طول رکوردها 180 و طول سکتورها 256 باشد و فاکتور بلاک 4 باشد چند درصد از شیار، بطور مؤثر قابل استفاده است؟

$$\frac{180 \times 4}{256 \times \boxed{?}} \times 100 = ✓$$

g ها

← کمترین مقداری که نرخ بزرگ تر که صدمت باشد

نرخ واقعی انتقال در دیسک

با این که یکی از عوامل هدر رفتن حافظه در دیسک گپ است ولی آن را در نظر نمی گیریم ~~G~~

1- وقتی تنها یک بلاک را به طور گزینشی بخواهیم بخوانیم.

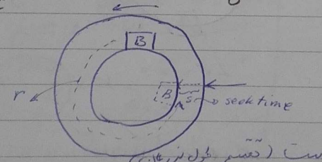
t: نرخ اسمی انتقال $t = \frac{B}{Btt}$

زمان خواندن یک بلاک (ms)

t_1: نرخ واقعی انتقال $t_1 = \frac{\overset{\text{بایت داده}}{B}}{\underbrace{S+r}_{\text{زمان تلفاتی}}+Btt}$

S: زمانی که طول می کشد که هد از یک شیار به شیار دیگر برود.

r: زمانی که طول می کشد تا B زیر هد قرار گیرد.



توجه: نرخ انتقال در حقیقت همان سرعت است (تقسیم طول بر زمان)

2- وقتی می خواهیم تعداد زیادی بلاک را بدون وقفه بخوانیم (نرخ انتقال انبوه)

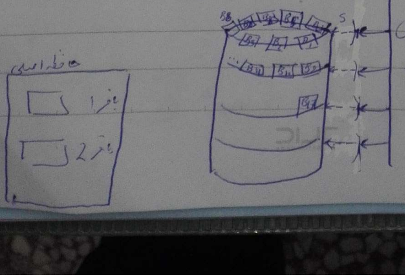
فرض 1) بافرینگ مضاعف داریم 2) تمام بلاک های مورد نظر روی یک سیلندر قرار گرفته اند
3) بافرها کارایی دارند

t_2: نرخ واقعی انتقال انبوه $t_2 = \frac{\overset{\text{خیلی}}{1000B}}{1S + \cancel{r} + 1000btt} = \frac{B}{btt}$

← یعنی cpu دو بلاک بافر ها را پردازش می کند ولی زمان آن کوچک تر از یک دور گردش دیسک است پس می تواند حین تمام شدن دو بلاک دیگر را پردازش کند

$CB < 2r$

زمان پردازش cpu روی بلاک در بافر



3- وقتی می‌خواهیم تعداد زیاد بلاک را بدون وقفه بخوانیم (نرخ انتقال انبوه)
با این تفاوت که بافرها کارایی ندارند.

فرض ۱) بافرینگ مضاعف داریم 2) تمام بلاک‌ها مورد نظر روی یک سیلندر قرار گرفته‌اند

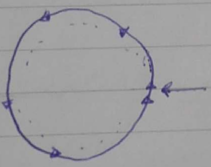
$$t_3 = \frac{2B}{s+\underbrace{2r}_{\text{زمان یک دور اضافه}}+2btt} = \frac{B}{\underbrace{\boxed{r}}_{\text{زمان یک نیم دور اضافه}}+btt}$$

(3) بافرها کارایی ندارد)

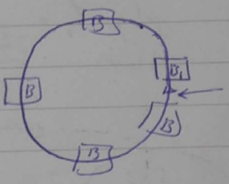
زمان دقیق درنگ دورانی

غیردقیق :

$$r = \frac{1}{2} \times \frac{\overset{\text{تبدیل به ثانیه}}{60} \times \overset{\text{تبدیل ms}}{1000}}{\underbrace{RPM}_{\text{تعداد دور در دقیقه}}}$$



$$\frac{2r+0}{2} = r$$



$$\frac{2r+0}{2} = r$$

T_f = track factor → تعداد بلاک‌های موجود در شیار

$$r' = \underbrace{r}_{\text{زمان رسیدن نقطه شروع خواندن و نوشتن زیر هد خواندن}} + \underbrace{r\,\frac{T_f-1}{T_f}}_{\text{زمان گردش دیسک برای رسیدن ابتدای بلاک زیر هد خواندن و نوشتن}}$$

3600

مثال: فرض نمایید تعداد دور های یک دیسک در هر دقیقه 1800 دور و تعداد بلاک های در هر شیار ترتیب فاکتور 5 باشد، مقدار دقیق درنگ دورانی را بدست آورید

$T_f = 5$

$Rpm = 3600$

$$r = \frac{1}{\cancel{2}} \times \frac{\cancel{60}\times 1000}{\cancel{3600}} = \frac{50}{6}$$

$$r' = \frac{50}{6} + \frac{\overset{10}{\cancel{50}}}{\cancel{6}_{3}} \cdot \frac{\overset{4}{5-1}}{\cancel{5}} = \frac{50}{6} + \frac{20}{3}$$

$$= \frac{50+40}{6} = \frac{90}{6} = \frac{30}{2} = 15\ ms$$

روش های افزایش نرخ انتقال داده

$$t = \frac{B}{s + r + btt}$$

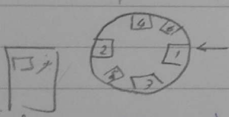
روش های کاهش s و r

1. روش های کاهش r

نرم افزاری

1. چیدمان بلاک ها روی شیار به صورت چند درمیان

اگر به جای 2 بافر به علت محدودیت حافظه یک بافر داشته باشیم در این حالت به جای اینکه در هر گردش 2 بلاک برود در 2 بافر، 1 بلاک میرود داخل 1 بافر پس برای افزایش کارایی به صورت زیر بلاک ها را می چینیم

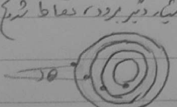


اگر دو بافر بود $3\times 2r$

$6\times 2r$

سخت افزاری

2. چیدمان نقاط شروع با زاویه (θ): در این روش برای جلوگیری از دور های اتلافی دیسک حاصل از زمانی که طول می کشد هد خواندن از یک شیار به شیار دیگر برود، نقاط شروع با زاویه تولید می کنند



$$\theta = \frac{360}{60\times 1000} \times Rpm \times S$$

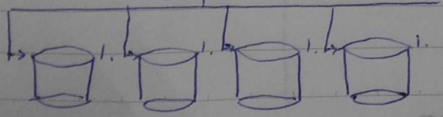
2. روش های کاهش S

سخت افزاری

1- استفاده از دیسک ها با بازوی ثابت

2- توزیع یک فایل روی چند دیسک بصورت همزمان

کنترل مرکزی



3600

مثال: فرض نمایید تعداد دورهای که دیسک در هر دقیقه می‌زند 1800 است و تعداد بلاک‌های موجود در هر شیار ترتیب گزینش 5 باشد، متوسط دقیق درنگ دورانی را بدست آورید

$T_f = 5$

$RPM = 3600$

$$r = \frac{1}{\cancel{2}} \times \frac{\cancel{60}\times \cancel{1000}}{\cancel{3600}} = \frac{50}{6}$$

$$r' = \frac{50}{6} + \frac{\cancel{50}}{\cancel{6}_3} \cdot \frac{\overset{4}{\overline{5-1}}}{\cancel{5}} = \frac{50}{6} + \frac{20}{3}$$

$$= \frac{50+40}{6} = \frac{90}{6} = \frac{30}{2} = 15 \text{ ms}$$

روش‌های افزایش نرخ انتقال واقعی

$$t = \frac{B}{s + r + btt}$$

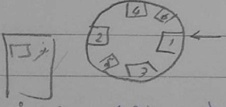
روش‌های کاهش s و r

1. روش‌های کاهش r

نرم افزاری

1. چیدمان بلاک‌ها روی شیار به صورت چند درمیان

اگر به جای 2 بافر به سبب محدودیت حافظه یک بافر داشته باشیم در این حالت به جای اینکه در هر گردش 2 بلاک برود در هر 2 بافر، 1 بلاک می‌شود داخل 1 بافر پس برای افزایش کارایی به صورت زیر بلاک‌ها را می‌چینیم

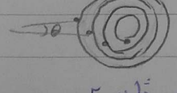


اگر دو بافر بود $3 \times 2r$

$6 \times 2r$

سخت افزار

2. چیدمان نقاط شروع با زاویه (θ): در این روش برای جلوگیری از دورهای باطل دیسک حاصل از زمانی که طول می‌کشد هد خواندن از یک شیار به شیار دیگر برود، نقاط شروع با زاویه تولید می‌شوند



$$\theta = \frac{360}{60 \times 1000} \times RPM \times S$$

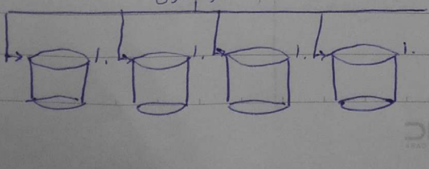
2. روش‌های کاهش S

سخت افزاری

1. استفاده از دیسک‌ها با بازوی ثابت

2. توزیع یک فایل روی چند دیسک به صورت همزمان

کنترل مرکزی

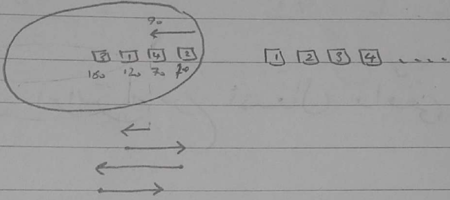


ادامه روش های کاهش S

3- استفاده از الگوریتم ها خاص حرکت بازوی خواندن و نوشتن

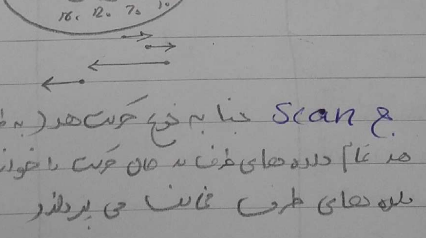
الف) FCFS (First come First service)

تصور کنید یک دیسک داریم که اطلاعات در سیلندر های مختلف قرار دارند. کاربران به ترتیب از شماره 1 تا ... در صف انتظار برای بازیابی اطلاعات دیسک است. هد خواندن و نوشتن ابتدا اطلاعات کاربر 1 را بازیابی کرده سپس کاربر 2 و به همین صورت تا آخر. در این روش هد لازم نیست تا آخرین سیلندر برود و برگردد


90
3
1
4
2
160
120
70
20
1
2
3
4
....

ب) SSTF

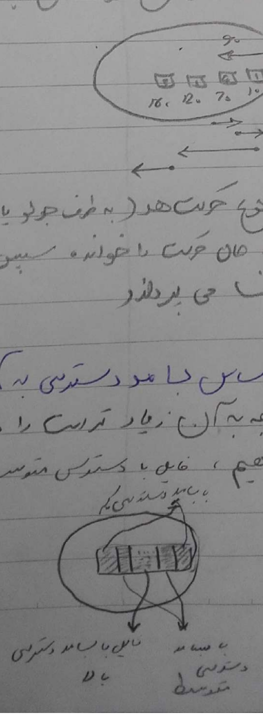
در این الگوریتم ابتدا به کاربری سرویس داده می شود که داده آن نزدیک تر به هد خواندن و نوشتن یا هم مسیر حرکت هد خواندن نوشتن باشد


90
1
160
120
70
10
1
2
3
4
...

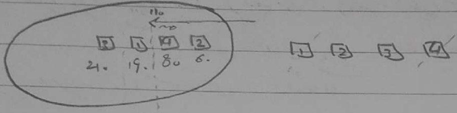
ج) Scan بنا به نوع حرکت هد (به طرف جلو یا عقب) هد تمام داده های طرف در حال حرکت را خوانده سپس به خواندن داده های طرف مخالف می پردازد

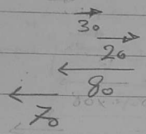
4- توزیع فایل ها براساس درصد دسترسی به آن ها

فایل هایی که تعداد مراجعه به آن زیاد تر است را در قسمت های میانی دیسک قرار می دهیم، فایل با دسترسی متوسط و کم در شکل مشخص شده اند


با درصد دسترسی کم
فایل با درصد دسترسی بالا
با درصد دسترسی متوسط

مثال: فرض کنید 4 درخواست 1 و 2 و 3 و 4 به ترتیب در سیلندرهای ۱۹۰، ۶۰، 210 و 80 قرار گرفته اند و فرض کنید که هد خواندن و نوشتن در لحظه با [illegible] روی سیلندر 110 قرار گرفته و به سمت شیارهای خارجی در حال حرکت است مشخص نمایید چه مدت زمانی طول می کشد تا هد خواندن و نوشتن به این 4 درخواست پاسخ دهد اگر بدانیم فاصله زمانی بین دو شیار متوالی ۰/۰۴ میلی ثانیه می باشد. (الگوریتم SSTF)





30
20
80
70

200 × ۰/۰۴ = 8ms

ساختار فایل‌ها

سازماندهی اطلاعات در یک قالب خاص را فایل گویند.

ضوابط طراحی فایل

۱. کاهش افزونگی داده ها (کم کردن فضا ذخیره سازی اطلاعات)

2. افزایش سرعت دسترسی به اطلاعات

3. سادگی در بهنگام سازی داده ها { درج / حذف / تغییر

4. افزایش امنیت داده ها
شامل چند تیم، پسورد، سطح دسترسی کاربر

5. اشتراکی شدن داده ها

6. (در متن کتاب)
⋮

انواع فایل های اولیه (کلاسیک)

1. فایل درهم pile

2. فایل ترتیبی

3. فایل ترتیبی شاخص دار

4. فایل مستقیم

5 - فایل چند ساخته

6 - فایل چند حلقه ای

ضوابط انتخاب فایل (کارآیی فایل)

1- متوسط اندازه یک رکورد (R)

2- زمان واکشی یک رکورد (T_F) fetch

3- زمان خواندن رکورد منطقی بعد (T_N) next

4- زمان درج یک رکورد (T_I) insert

5- زمان تغییر محتوای رکورد (T_U) update

6- زمان خواندن کل فایل (T_X)

7- زمان سازماندهی مجدد فایل (T_Y)

① متوسط اندازه یک رکورد

حالات فایل:

1- فایل پراکنده (sparse): فایلی که بعضی از رکوردها آن (بعضی از فیلدهایش) دارای مقدار نیستند.

ID	name	L.nam	Addr
100	—	—	تهران
200	—	—	
300	—	—	رشت
400	—	—	مشهد
500	—	—	

ID(3)
Name(10)
Lname(15)
Addr(15)

43

2. فایل متراکم Dense : فایلی که تمام فیلدهای رکوردهای آن پر است

3. فایل دارای افزونگی : فایلی که بعضی از فیلدها در بعضی از رکوردها مقدار تکراری باشند.

توجه: یک فایل هم ممکن است پراکنده باشد و دارای افزونگی هم باشد. همچنین ممکن است متراکم باشد و دارای

انواع افزونگی
- طبیعی : از روی اجبار.
- تکنیکی : از روی اختیار.

توجه: در ساختار DataBase همواره سعی می‌شود که از تکرار کاسته شود.

ضوابط کارایی یا انتخاب فایل

R : متوسط اندازه یک رکورد

حالات فایل
1. فایل پراکنده
2. فایل متراکم
3. فایل دارای افزونگی
   - طبیعی (اجبار)
   - تکنیکی (اختیار)

مثال برای افزونگی طبیعی:

درس دوم	درس اول	تحصیل	نام	شماره
هندسی [illegible]	ذخیره 1918			100
	ذخیره 1918			200
هندسی [illegible]				300
	ذخیره 1918			400

در اینجا ما به اجبار نام درس ذخیره را تکرار کردیم

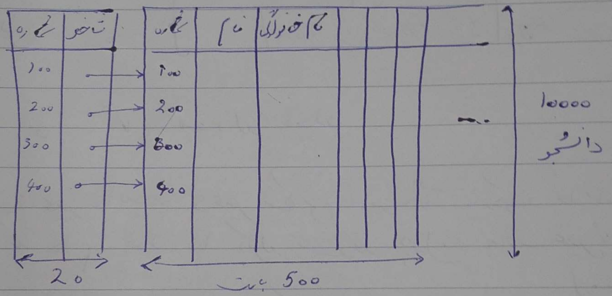
برای جلوگیری از کاهش کاراکترها می‌توان از کد استفاده کرد

روش دیگر برای کاهش فضای هرز در افزونگی طبیعی ماتریس بیتی است

.....	زن	مرد	نام خانوادگی	نام	شماره
		1			
		1			
	1				
	1				
		1			
	1	1			

مثال برای افزونگی تکنیکی :

برای ایجاد شاخص و بالا بردن سرعت جستجو باید فایل شاخص استفاده شود که خود مقادیر تکراری شماره ها دارد. ایجاد این فایل برای افزایش سرعت است و کاملاً اختیاری است


شماره
شاخص
شماره
نام
نام خانوادگی
100
200
500
400
100
200
500
400
.....
10000
دانشجو
20
500 بایت

حجم فایل دانشجو : $500 \times 10000 = 5MB$

" " شاخص : $20 \times 10000 = 200KB$

R_1 R_2

$R = 20 + 500 = 520$

زمان واکشی یک رکورد T_F

زمان واکشی یک رکورد بستگی به روش جستجوی یک رکورد دارد

روش جستجوی یک رکورد در فایل :

1. روش خطی سرچ در همه سطر ها ابتدا به دنبال آنها جستجو (ID=400) می گردد

2. روش دودویی (Binary) سرچ ← ترتیبی

R1 | A1 = V1 , A2 = V2
R2 | A3 = V3
R3 | A4 = V4 , A1 = V5 , A3 = V6

غالب غیر ثابت و غیر مکانی

A : اسم فیلد و V : مقدار فیلد

در رکورد های با غالب غیر ثابت و غیر مکانی برخلاف رکورد ثابت و مکانی؛ همواره باید نام فیلد ها تکرار شوند چون غالب غیر ثابت است.

ولی در غالب ثابت و مکانی احتیاجی به تکرار نام فیلد ها نیست چون غالب ثابت و مکانی است

فیلدها (مکانی)

ID	Nam	Lnam
1..	—	—
2..	—	—
3..	—	—
4..	—	—

رکورد ثابت و مکانی

توجه: زمانی مورد استفاده قرار می گیرد که ① فیلدی نمی توانند به خود نظم بگیرد و یا احتیاجی به نظم ندارد.

② امنیت اطلاعات برایمان مهم باشد، زیرا هکر ها هک کرد نمی توانند چینش فیلد ها را حدس بزنند

توضیح. جستجو و خواندن فایل خطی است: رکوردها را یکی یکی با [illegible] مقایسه می کند.

اندازه گیری ضوابط در فایل برهم

مفروضات:

1- n تعداد رکوردهای فایل در زمان ایجاد

2- o تعداد رکوردهای اضافه شده به فایل

3- d تعداد رکوردهای حذف شده از فایل

4- A متوسط اسم یک فیلد

برای مثال $\underbrace{ID}_{2}=123\ ,\ \underbrace{Name}_{4}=Ali\ ,\ \underbrace{LName}_{6}:Ahmadi$

$$A=\frac{12}{3}=4$$

5- V متوسط مقدار یک فیلد

$ID=\underbrace{123}_{3}\ ,\ Name=\underbrace{Ali}_{3}\ ,\ L\text{-}name:\underbrace{mohammadi}_{9}$

$$V=\frac{15}{3}=5$$

6- a تعداد فیلدهای فایل

$A_1=V_1\ ,\ A_2=V_2$	
$A_3=V_3$	
$A_4=V_4\ ,\ A_1=V_5\ ,\ A_3=V_6$	

$$a=\frac{6}{3}=2$$

$$\underset{\text{متوسط اندازه رکورد}}{R}=a\,(\underbrace{A+V+2}_{\text{متوسط اندازه یک فیلد}})$$

$2 \rightarrow$ = و ،

$$T_F=\begin{cases}A_1=V_1 \simeq 0\\ A_1=V_5 \simeq \underset{\text{زمان خواندن کل فایل}}{T_x}\end{cases}\ \frac{0+T_x}{2}=\frac{1}{2}T_x \Rightarrow T_F=\frac{1}{2}T_x$$

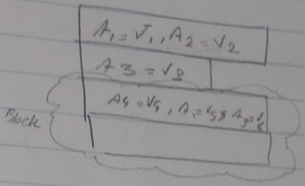
$$\underset{\text{زمان خواندن رکورد منطقی بعدی}}{T_N}:T_F$$

زمان درج یک رکورد

$$T_I = \underbrace{(s+r+btt)}_{\text{زمان پیدا کردن آخرین بلاک فایل}} + c_B + \underset{2r}{T_{RW}} = s+3r+btt$$

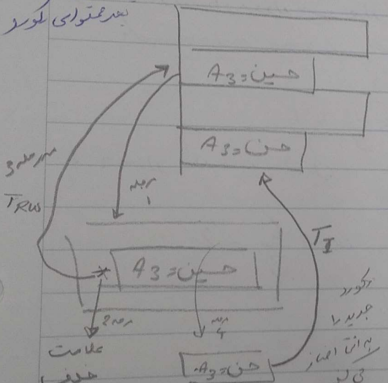
cpu رکورد مدت زمانی که طول می کشد جدید را به انتهای اضافه کند

چون رکوردها هیچ وابستگی منطقی به یکدیگر ندارند، انتهای فایل اضافه می کنیم


A1 = V1 , A2 = V2
A3 = V3
A4 = V4 , A1 = V5 A3
Block

زمان تغییر محتوای رکورد

$$T_a = T_F + T_{RW} + T_I$$


A3 = حسین
A3 = حسن
مرحله 1
مرحله 2
مرحله 3
T_RW
مرحله 4
T_I
A3 = حسین
علامت حذف
A3 = حسن
رکورد جدید را به انتهای اضافه می کنیم

علامت حذف رو پاک می کنیم و حسین را تبدیل به حسن می کنیم

$$T_X = (n+o)\frac{R}{\boxed{t}}$$

سرعت خواندن گسترده

زمان خواندن ≃ زمان نوشتن

$$T_Y = (n+o)\frac{R}{t} + (n+o-d)\frac{R}{t}$$