فایل با ساختار پایل یا برهم:

این فایل ساختاری دارد فاقد هرگونه نظم، یعنی رکوردها بر اساس مقادیر هیچ صفت خاصه ای مرتب نیستند. در ایجاد این فایل، هیچگونه پیش پردازشی، از قبیل تحلیل داده ها، بررسی آماری، رده بندی یا بخش بندی رکوردها، صورت نمی گیرد، در بهترین حالت، نظم بین رکوردها، نظمی است زمانی، انگار رکوردها بر یکدیگر پشته شده باشند. این ساختار فاقد هرگونه استراتژی دستیابی است.

رکوردها طول متغیر دارند. تعداد صفات خاصه و نیز مکان فیلدهای متناظر با صفات خاصه، در نمونه های مختلف رکوردها، متفاوت است.

فایل به صورت زیر است:

: اسم صفت خاصه

: مقدار صفت خاصه

به عبارت دیگر، قالب رکورد، طرح غیر ثابت و غیر مکانی دارد.

برای محیطهای عملیاتی که در آن داده ها، اساسا نظم پذیر نباشند و پیش پردازش در آنها انجام نشده باشد و در برخی از محیطهای نظامی

و به عنوان یک ساختار مبنائی در مطالعه بقیه ساختارها به کار می آید.

ارزیابی کارایی:

متوسط اندازه رکورد:

- فایل در لوله اولیه، n رکورد دارد.
- کل تعداد صفات خاصه در نظر گرفته شده در محیط عملیاتی را a می نامیم.
- متوسط تعداد صفات خاصه در یک رکورد را با a' نشان می دهیم.
- متوسط حافظه لازم برای اسم صفت خاصه را، A بایت در نظر می گیریم.

- متوسط حافظه لازم برای مقدار صفت خاصه را v بایت فرض می کنیم.

زمان واکنشی یک رکورد:

آرگومان جستجو در درخواست به صورت K=V داده می شود.

عملیات لازم: خواندن بلاک حاوی رکورد مورد نظر. اما به دلیل بی نظم بودن، رکورد مورد نظر ممکن است در اولین بلاک فایل باشد یا مثلاً در آخرین (و یا در هر بلاک دیگر). بنابراین بطور متوسط نصف بلاکهای فایل باید خوانده شود. اگر تعداد بلاکهای فایلb باشد و هر بلاک بطول B بایت، زمان واکنشی از این رابطه بدست می آید:

چون تعدادی بلاک باید خوانده شوند، لذا نرخ انتقال انبوه در نظر گرفته شده است.

زمان به دست آوردن رکورد بعدی

زمان عمل درج

- خواندن آخرین بلاک فایل.
- کار در بافر (که زمانش را در ارزیابی دخالت نمی دهیم).
- بازنویسی بلاک.

زمان بهنگام سازی از طریق تغییر

- واکشی رکورد بهنگام درآمدنی
- ضبط نشانگر حذف شده در نسخه قدیم
- ایجاد نسخه جدید

- بازنویسی نسخه قدیم
- درج نسخه جدید در انتهای فایل

: واکنشی رکورد بهنگام درآمدنی

: بازنویسی همین رکورد با نشانگر حذف شده

: درج نسخه جدید

فایل با ساختار ترتیبی:

مقدمه و معرفی ساختار:

این فایل نسبت به فایل پایل دو بهبود ساختاری دارد:

۱) در لود اولیه، تمام نمونه رکوردها بر اساس مقادیر یکی از صفات خاصه منظم هستند و این نظم با هم جواری فیزیکی پیاده سازی می شود. وجود کلید با خاصیت یکتائی، تضمین است زیرا در بدترین حالت با ترکیب تمام صفات خاصه یک رکورد، می توان به مقدار مرکب یکتا رسید و اگر نرسیم معنایش این است که بر نمونه موجودیت در واقع یکی بوده است و می گوئیم فایل، افزونگی از نظر تکرار رکورد در فایل دارد. گاهی نیز به هر نمونه رکورد یک شماره واحد داده می شود که در این صورت به آن کلید خارجی رکورد می گوئیم.

۲) تمام نمونه رکوردها از قالب از پیش طراحی شده ای برخوردارند.

فایل نسبت به فایل پایل، انعطاف پذیری اش را از دست می دهد.

در عمل درج، منطقاً باید نقطه درج را پیدا کرد و درج رکورد در انتهای فایل، نظم را به هم می زند. همچنین وجود قالب از پیش تعریف شده با طول ثابت موجب کاهش انعطاف پذیری فایل در عمل بهنگام سازی می شود، مثلاً طول رکوردها نمی تواند در بهنگام سازی، تغییر کند.

رکورد روی رسانه ذخیره سازی، عملا تصویری باشد از آنچه که در حافظه اصلی ساخته می شود

عدم تقارن در فایل ترتیبی:

فایل ترتیبی دارای عدم تقارن است، زیرا همینکه فایل را روی یک صفت خاصه (ساده یا مرکب) منظم کنیم، این نظم یک استراتژی دستیابی را برای فایل تأمین می کند، متکی بر آن صفت خاصه و در نتیجه صفات خاصه دیگر چنین ویژگی را ندارند و نقشی نخواهند داشت در عملیات روی فایل. به عبارت دیگر، صفت خاصه نظم، کلید فایل است و بر هر صفت خاصه دیگر رجحان دارد.

موارد استفاده:

عمدتاً در کاربردهای تجاری، وقتی که با سیستم یکجا (Batch) پردازش می کنیم و بطور کلی وقتی که ماهیت پردازش محیط عملیاتی، ترتیبی باشد.

اصطلاحاً می گوییم این فایلها (که باید ادغام شوند) فایلهای همتوالی (co-sequential) می شوند.

جستجوی باینری

جستجوی باینری، در یک محیط منظم خارجی باید در دو سطح انجام گیرد، در سطح اول، جستجویی در فایل داریم تا بلاک مورد نظر پیدا شود (واحد جستجو در این سطح، بلاک است). برای این کار طبعاً باید بلاکها خوانده شوند. در سطح دوم برای هر بلاک که به بافر آورده می شود، یک جستجوی باینری درون بلاکی داریم. این هر دو جستجو در ارزیابی زمان دخالت دارند.

در بررسی محتوای یک بلاک، کلید اولین و آخرین رکورد بلاک بررسی می شود تا مشخص شود که رکورد مورد نظر در بلاک هست یا نه. و اگر در بلاک وجود نداشت، بلاک میانی دیگر باید خوانده شود.

در این روش، کلید رکورد مورد نظر (آرگومان جستجو) با کلید آخرین رکورد هر بلاک مقایسه می شود، چنانچه این کلید از آرگومان جستجو بزرگتر باشد، محتوای بلاک مورد جستجو، مثلا به صورت خطی بررسی می شود. در واقع سیستم فایل با نوعی پرش از بلاکها، بلاکی را که باید محتوایش بررسی شود، به دست می آورد.

در این ساختار رکورد بعدی، معمولا بلافاصله در دسترس است، زیرا بسیار محتمل است که در همان بلاکی باشد که رکورد فعلی. بنابراین می توان زمان لازم را

در حالت خاص ممکن است رکورد بعدی در بلاکی دیگر باشد. در این حالت آن بلاک نیز باید خوانده شود.

رکورد درج شدنی را نمی توان به انتهای فایل اصلی اضافه کرد. زیرا منطقاً باید نقطه درج به دست آید و این کار زمانگیر است. لذا در محیطهای با فایل ترتیبی، فایل جداگانه تراکنش ها وجود دارد و رکوردها در آن درج می شوند تا در سازماندهی مجدد، فایل بر اساس نظم مورد نظر، بازآرایی شود. برای ارزیابی زمان درج، دو حالت را در نظر می گیریم:

درج در فایلهای کوچک

می توان در فایلهای کوچک، نقطه درج را بازشناسی، رکورد را در آن نقطه درج کرد. در این صورت بقیه رکوردها باید به سمت پایان فایل (EOF) شیفت داده شوند. چون مشخص نیست که نقطه درج کجا است، لذا دقیقاً مشخص نیست چه تعداد بلاک باید شیفت داده شود، لذا بطور متوسط، نصف بلاکهیا فایل شیفت داده می شوند. پس عملیات لازم در این حالت عبارتست از:

- یافتن نقطه منطقی درج.
- درج رکورد در بلاکهای مورد نظر (کار در بافر).
- شیفت دادن بلاکها از نقطه منطقی درج به سمت EOF.

و خواهیم داشت:

رکورد درج شدنی در آخرین بلاک فایل تراکنش ها درج می شود و این عمل زمان درج در پایل را لازم دارد.

زمان بهنگام سازی تغییر دهنده:

اگر مقدار کلید عوض نشود، می توان رکورد را بصورت درجا بهنگام درآورد ولی معمولا برای بهنگام سازی نیز از فایل تراکنش ها استفاده می شود.

عملیات لازم:

- واکشی رکورد بهنگام درآمدنی.
- عمل بهنگام سازی در بافر (ایجاد نسخه جدید).
- درج رکورد بهنگام درآمده، همراه با یک رکورد کوچک منضم به آن.

فایل تراکنش ها باید مرتب شود (البته اگر بخواهیم خواندن تمام فایل با رعایت سریالیتی در هر یک از دو فایل باشد). زمان چنین برآورد می شود:

اگر بخواهیم تمام فایل را با سریالیتی کامل بخوانیم باید فایل اصلی و فایل تراکنش در هم ادغام شوند یعنی عملاً سازماندهی مجدد صورت گیرد.

۱- خواندن فایل اصلی.
۲- مرتب کردن فایل تراکنش (تا با فایل اصلی همتوالی شود).
۳- خواندن فایل تراکنش.
٤- ادغام دو فایل.
٥- بازنویسی با خارج کردن حذف شدنی های ناشی از عمل حذف یا عمل بهنگام سازی.

در این ساختار، فایل داده یی، در لود اولیه، یک فایل ترتیبی است. یعنی در لود اولیه، استراتژی دستیابی بر اساس نظم کلیدی تأمین است. به علاوه، برای تسریع در واکشی رکوردها، ساختار شاخص به آن منضم می شود.
عناصر تشکیل دهنده این ساختار عبارتند از:
۱- فایل ترتیبی که اصطلاحاً به آن ناحیه اصلی می گویند (Primery Area).
۲- شاخص.
۳- نشانه روها (برای ایجاد شاخص و نیز برقراری ارتباط بین رکوردها).
٤- ناحیه سرریزی (overflow area) برای انجام عملیات ذخیره سازی در فایل پس از لود اولیه.
وجود شاخص، نوعی استراتژی دستیابی مستقیم را برای فایل تأمین می کند، پس این ساختار دو استراتژی دستیابی دارد: ترتیبی روی مقادیر صفت خاصه کلید و مستقیم از طریق شاخص بندی.
نحوه پردازش فایل در بازیابی:
الف) خواندن فایل به صورت پی در پی (بازیابی تمام رکوردها): ناحیه اصلی و ناحیه سرریزی به صورت بلاک به بلاک خوانده می شوند و چون، به شرحی که

خواهیم دید، رکوردهای ناحیه سرریزی دارای سریالیتی با هم جواری فیزیکی نیستند، لذا در این نحو خواندن، سریالیتی وجود ندارد.

ب) خواندن فایل به صورت سریال: بلاکهای ناحیه اصلی روی کلید خوانده می شوند تا به رکوردی برسیم که دارای یک نشانه رو به ناحیه سرریزی است. از این پس، به ناحیه سرریزی رفته، خواندن را با طی کردن زنجیره سرریزی ها، دنبال می کنیم تا به پایان زنجیره برسیم و مجدداً به ناحیه اصلی بازگشت می کنیم (به نقطه گسست) و بدین نحو خواندن سریال ادامه می یابد. (شکل ٤-١٥)

به نحوی که خواهیم دید، از ناحیه اصلی، تعدادی ذخیره داریم ادامه یافته در ناحیه سرریزی و همین زنجیره ها، پردازش سریال را امکان پذیر می سازند و نیز در واکشی یک رکورد، چنانچه رکورد مورد نظر در ناحیه سرریزی باشد، مورد پیمایش قرار می گیرند.

ج) بازیابی یک رکورد از روی کلید: برای بازیابی یک رکورد، باید در فایل شاخص جستجو کرد و با توجه به مدخل مربوطه در فایل شاخص، در نهایت به رکورد مورد نظر رسید و البته نحوه عمل بستگی به ساختار شاخص دارد. ممکن است مدخل (Entry) شاخص لزوماً ما را به رکورد مورد نظر نرساند، بلکه به قسمتی از فایل برسیم که رکورد در آن باشد و چه بسا که از آن قسمت هدایت شویم به ناحیه سرریزی.

شکل ٤-١٦: نمایش کلی نحوه عمل است.

شاخص:

تعریف شاخص و برخی اصطلاحات

شاخص یا فایل شاخص، مجموعه ایست از تعدادی مدخل. هر مدخل مربوط است به یک رکورد داده یی و یا یک گروه از رکوردهای داده یی و برای دستیابی مستقیم بکار می آید. ساختمان یک مدخل شاخص مطابق شکل (٤-١٧) است.

خود فایل شاخص بر اساس مقادیر صفت خاصه نظم، منظم است. معمولا صفت خاصه شاخص، همان صفت خاصه ای است که فایل داده یی اصلی بر آن اساس منظم است و در این حالت می گوئیم شاخص اصلی داریم (Primary Index). اما می توان بر یک فایل روی صفات خاصه دیگر نیز شاخص داشت، موسوم به شاخص ثانوی و ... (Secondary Index). پس فایل شاخص، اساساً فایلی است با رکوردهای کوچک (بایت v+p) و همتوالی با ناحیه اصلی.

اگر خود فایل شاخص بزرگ باشد، طبعاً طی کردن ترتیبی آن زمانگیر می شود. برای اجتناب از زمانگیر شدن عمل جستجو در فایل شاخص، دو راه وجود دارد (در این ساختار مبنایی):

راه اول: اعمال یک الگوریتم مناسب جستجو در فایل شاخص، مثلا جستجوی باینری یا جستجوی بلاکی.

راه دوم (که رایج است): ایجاد سطوح مختلف شاخص (شاخص چند سطحی). تعداد سطوح شاخص را به x نشان می دهیم و می گوئیم شاخص سطح اول، شاخص سطح دوم و ... و شاخص سطح x، که بالاترین است. شاخص سطح x از نظر اندازه چنان است که مقیم (Resident) کردن آن در حافظه اصلی به صرفه می باشد (کمتر از یک بلاک)، و به آن می گوئیم سر شاخص (Master Index).

تعریف لنگرگاه: (Anchor point)

هر نقطه از ناحیه اصلی که از مدخل شاخص به آن نشانه رو داریم. لنگرگاه ممکن است هر یک از رکوردهای ناحیه اصلی باشد و یا رکوردی از هر گروه رکوردها. از این نقطه نظر، دو نوع شاخص داریم:

الف) شاخص متراکم (Dense)

اگر در سطح اول شاخص، به تمام رکوردها، نشانه رو داشته باشیم، می گوییم شاخص متراکم است و لنگرگاه رکوردی.

ب) شاخص غیر متراکم (Non Dense)

اگر به هر گاه رکوردها، (مثلا یک بلاک) در سطح اول، نشانه رو داشته باشیم، شاخص را غیر متراکم گوئیم و لنگرگاه در این حالت گروهی است.

تعریف ظرفیت نشانه روی شاخص (Index fanout)

فایل شاخص نیز، مثل فایل داده یی، بلاک بندی شده است. تعداد مدخلهای یک بلاک شاخص را ظرفیت نشانه روی آن می گویند. در واقع همان فاکتور بلاک بندی است برای بلاک شاخص و با پارامتر y آنرا نمایش می دهند.

چگونگی انتخاب گروه رکوردها (شاخص غیر متراکم):

گروه می تواند مستقل از تقسیمات سخت افزاری رسانه باشد و یا وابسته به آن. در حالت نخست، شاخص بندی را مستقل از سخت افزار و اصطلاحاً نرم افزاری گویند و گروه معمولا همان بلاک است و گاهی چندین بلاک، با نام خاص، مثلا در سیستم VSAM، گروه از چند بلاک تشکیل شده است و دو نوع گروه بندی وجود دارد، به نامهای فاصله کنترل و ناحیه کنترل (CI,CA) که در فصل مربوطه خواهیم دید. در حالت دوم، شاخص بندی را اصطلاحاً سخت افزاری می گویند. در این نحو گروه بندی، تقسیمات دیسک، یعنی سکتور، شیار، استوانه، و در صورت لزوم، خود دیسک پک (volum) به عنوان گروه انتخاب می شود، مثل سیستم IBM/SAM یا CDC/SIS، مطابق شکل ٤-١٨. گاه فقط به ایجاد شاخص استوانه و شاخص بلاک ها در استوانه، بسنده می کنند.

در شاخص غیر متراکم، لنگرگاه می تواند اولین رکورد هر گروه باشد و یا آخرین رکورد آن. بدیهی است که شرط ایجاد شاخص غیر متراکم این است که رکوردهای داده یی منظم باشند (مرتب شده روی صفت خاصه شاخص). در

صورت عدم وجود نظم بین رکوردها، باید شاخص متراکم ایجاد کرد، به نحوی که در ساختار چهارم خواهیم دید.

شرح مثال: رکوردهای داده یی روی شماره بیمه های اجتماعی کارمندان مرتب هستند. در هر بلاک داده یی، سه رکورد داریم. در سطح اول شاخص، برای هر بلاک یک مدخل شاخص ایجاد شده است (گروه، در این مثال بلاک است) و شاخص غیر متراکم. در هر بلاک شاخص، هشت مدخل داریم (y=۸). در فیلد v از مدخل شاخص، کوچکترین مقدار کلید رکوردهای هر بلاک جای دارد، همانگونه که در شکل ٤-۱۹ مشهود است. در سطح اول دو بلاک شاخص داریم و یک سطح دیگر ایجاد شده است، پس x=۲.

توجه داریم که در این ساختار، شاخص به ناحیه سرریزی ناظر نیست، فقط برای ناحیه اصلی ایجاد می شود و از این نقطه نظر می گوئیم که شاخص در این ساختار حالت ایستا دارد (فاقد پویایی است، بر خلاف شاخص در ساختار چهارم که خواهیم دید).

مثالی از شاخص بندی نرم افزاری:

فایلی داریم با یک میلیون رکورد. با توجه به مفروضات زیر، مطلوبست طراحی ساختار شاخص برای این فایل.

حل:

طول یک مدخل شاخص

تعداد مدخلها در سطح اول

حافظه مصرفی برای سطح اول شاخص

حافظه مصرفی زیاد است، لذا سطح دوم شاخص را ایجاد می کنیم:

تعداد بلاکهای سطح اول
برای نگهداری در حافظه اصلی، زیاد است.
تعداد بلاکها در سطح دوم

پس ساختار شاخص را در سه سطح ایجاد می کنیم.
محاسبه تعداد سطوح:
داریم:

و تعداد سطوح شاخص از رابطه زیر بدست می آید:

و یا:

در مثال دیده شده:

هر چه تعداد سطوح بیشتر باشد، دفعات دستیابی برای واکشی رکورد بیشتر خواهد بود. در نتیجه، برای کاهش تعداد سطوح، باید ظرفیت نشانه روی شاخص را افزایش داد. و لازمه این کار، داشتن بلاک شاخص طولانی تر و کوتاه تر کردن طول مدخل شاخص است. ولی اندازه بلاک شاخص همانست که اندازه بلاک داده یی و انتخاب آن، به امکانات بافرینگ سیستم بستگی دارد.

محل نگاهداری فایل شاخص:

دیدیم که شاخص، خود یک فایل است با ساختار درونی خاص خود و رکوردهایی به طول v+p بایت. معمولا شاخص بالاترین سطح، موسوم به سر شاخص (Master index) در حافظه اصلی نگاهداری می شود و بلاکهای سایر سطوح شاخص روی دیسک، در شیارهای آغازین هر استوانه.

بررسی مسأله سرریزی:

جنبه های مختلف مسئله

در بررسی مسئله سرریزی باید به سوالات زیر پاسخ داد:

۱) فضای لازم برای درج رکوردهای سرریزی چگونه انتخاب شود؟

۲) فضای انتخاب شده، چگونه در محیط فیزیکی (روی دیسک) به فایل تخصیص یابد؟

۳) عمل درج سرریزی ها با چه تکنیکی انجام شود و سریالیتی رکوردها چگونه تأمین گردد؟

در پاسخ به سوال اول، سه راه حل به نظر می رسد:

الف) در نظر گرفتن جا در هر بلاک در لود اولیه (چگالی لود اولیه کوچکتر از یک باشد).

ب) ایجاد یک فایل جداگانه (مثل ساختار دوم).

ج) در نظر گرفتن ناحیه ای جداگانه در همان فایل داده یی.

راه حل الف) هر چند به نظر می رسد که از نظر قوی بودن لوکالیتی رکوردها، راه خوبی باشد، زیرا رکوردهای سرریزی در همان بلاک رکوردهای اصلی درج می شوند، ولی اگر توزیع سرریزی ها نایکنواخت باشد، برخی بلاکها با کمبود جا برای درج سرریزی ها مواجه می شوند (زیرا پر می شوند) و برخی دیگر پر نشده باقی می مانند و برای بلاکهای پر شده باز هم مسئله سرریزی ها مطرح خواهد بود.

راه حل ب) راه حل زمانگیری است، زیرا عملیات اساساً در فایل جداگانه ای انجام می شود و با توجه به اینکه باید بین رکوردهای فایل اصلی و رکوردهای سرریزی ارتباط ایجاد شود (به کمک نشانه روها)، ایجاد این ارتباط بین دو فایل جداگانه، از ایجاد ارتباط بین رکوردهای یک فایل واحد، دشوارتر خواهد بود و ضمناً لوکالیتی رکوردهای سرریزی ضعیف می شود.

راه حل ج) مناسب ترین راه حل در این ساختار است. رکوردهای سرریزی با استفاده از تکنیک هایی که خواهیم دید، درج خواهند شد.

در پاسخ به سوال دوم، دو راه وجود دارد:

۱- تخصیص استوانه هایی در انتهای فایل، برای ایجاد ناحیه جداگانه.

۲- تخصیص شیارهایی در انتهای هر استوانه، به عنوان ناحیه سرریزی همان استوانه.

راه حل اول در شکل ٤-۲۱ نمایش داده شده است:

این راه حل مناسب نیست، زیرا سبب می شود که لوکالیتی رکوردهای سرریزی ضعیف شود و در نتیجه متوسط زمان استوانه جویی افزایش یابد.

راه حل دوم در شکل ٤-۲۲ دیده می شود:

این راه حل: متوسط زمان استوانه جویی را کاهش می دهد، زیرا رکوردهای سرریزی هر استوانه در همان استوانه جای دارند. البته وقتی که ناحیه سرریزی یک استوانه پر شود، ناحیه دیگری برای درج سرریزی ها باید ایجاد کرد (ناحیه سرریزی اولیه و ثانویه) و یا اینکه فایل را سازماندهی مجدد کرد.

شکل ٤-۲۳ وضعیت کلی رکوردهای یک استوانه، و رکوردهای اصلی و سرریزی آن را نشان می دهد.

در راه حل اخیر این سوال مطرح می شود که اندازه ناحیه سرریزی در هر استوانه، چه باید باشد و آیا در تمام استوانه ها، این اندازه یکسان است. منطقا نیازی به مساوی بودن اندازه ناحیه سرریزی در هر استوانه نیست و بستگی دارد به حجم عملیات درج در هر استوانه. اگر بتوان حجم این عملیات را ارزیابی کرد، آنگاه ممکن است متناسب با آن، اندازه ناحیه را در هر استوانه انتخاب کرد. ولی معمولا اندازه ناحیه سرریزی در تمام استوانه ها یکسان گرفته می شود و البته امکان دارد در صورت تا یکنواخت بودن توزیع درجی ها، بعضی از استوانه

ها با کمبود جا مواجه شوند و به ناحیه سرریزی ثانوی نیاز داشته باشند، در حالیکه در برخی دیگر از استوانه ها، ناحیه سرریزی پر نشود و یا اینکه اساساً درجی در آنها صورت نگیرد.

در پاسخ به سوال سوم، تکنیک های موجود را مطرح می کنیم:

تکنیک های درج سرریزی

دو تکنیک وجود دارد:

۱) درج در اولین بلاک جادار در ناحیه سرریزی

۲) درج با push through

درج در اولین بلاک جادار در ناحیه سرریزی

در این تکنیک، رکورد جدید، مستقیماً وارد بلاکی از ناحیه سرریزی می شود و در اولین مکان آزاد جای می گیرد (در اولین بلاک جادار). سپس از رکورد منطقاً پیشین به رکورد درج شده نشانه رو ایجاد می شود و به ترتیب زنجیره رکوردهای سرریزی پدید می آید. در این تکنیک برای هر رکورد از ناحیه اصلی و ناحیه سرریزی یک فیلد نشانه رو وجود دارد. البته ممکن است در بعضی از رکوردها، محتوای این فیلد Null باشد.

در فیلد نشانه رو آخرین رکورد هر زنجیره، Null داریم به معنای اینکه پایان زنجیره است. پس در این روش، زنجیره هایی داریم که مبدأ آنها، نقاطی از ناحیه اصلی است و ادامه دارند در ناحیه سرریزی.

بطوریکه در این مثال دیده می شود، یکی از زنجیره ها چنین است:

تکنیک درج با Push through

در این تکنیک سعی می شود که سریالیتی بلاک ها در ناحیه اصلی حفظ شود. رکورد جدید در بلاک مربوطه در ناحیه اصلی، در محلی که منطقاً باید جای گیرد، وارد می شود، بعد از رکورد منطقا قبلی اش و رکوردهای بعدی همان بلاک (البته

غیر از اولین رکورد بلاک) به سمت انتهای بلاک شیفت داده می شوند و در نتیجه رکورد آخر بلاک push می شود به اولین بلاک جادار در ناحیه سرریزی. البته زنجیره رکوردهای سرریزی هم ایجاد می گردد. مثالی از این تکنیک در شکل ٤-٢٦ آمده است. بطوریکه در این مثال مشاهده می شود، یکی از زنجیره ها چنین است:

در این روش، برای هر بلاک از ناحیه سرریزی یک نشانه رو داریم (و نه برای هر رکورد).

ضمناً ممکن است، رکوردهای بهم زنجیر شده، در بدترین حالت، هر یک در یک بلاک از ناحیه سرریزی باشد، در مثال، رکوردهای با کلید 128 و 129 در یک بلاک جای دارند و رکورد 075 در بلاکی دیگر. طول زنجیره در این روش بیشتر است از روش قبلی، ولی در عوض پردازش سریال فایل تسهیل می گردد. وضعیت نهایی فایل در شکل ٤-٢٦ دیده می شود.

در هیچیک از دو تکنیک، فایل شاخص تغییر نمی کند، زیرا در این ساختار شاخص فقط ناظر است به ناحیه اصلی. به عبارت دیگر، ساختار شاخص، حالت پویا ندارد.

موارد استفاده ساختار:

این ساختار در محیطهایی به کار می رود که در آنها نیاز به پردازش سریال فایل روی یکی از صفات خاصه (کلید) مطرح بوده، به علاوه واکشی تک رکوردها از طریق مقدار کلید آنها عمل رایجی باشد، در اغلب سیستمهای تجاری - مدیریتی، این ساختار مورد استفاده قرار می گیرد، البته گاه در شکل کاملترش که خواهیم دید.

ارزیابی کارایی:

از آنجا که سیستمهای پیاده سازی شده بر اساس این ساختار، جنبه های تکنیکی گوناگون دارند (مثلا ISAM با SIS تفاوت هایی دارد)، لذا برای ارزیابی، طرحی را در نظر می گیریم که رایجترین است، با مفروضات زیر:

سطح اول شاخص غیر متراکم است.

فایل شاخص همتوالی است با فایل داده یی.

بلاک های شاخص در یک استوانه جای دارند و بلاکهای ناحیه سرریزی یک استوانه نیز در همان استوانه قرار دارند.

تکنیک درج Push through است.

ساختار شاخص، حالت پویا ندارد.

بعد از سازماندهی مجدد، بلاکهای شاخص و ناحیه اصلی پر هستند و بلاکهای ناحیه سرریزی خالی.

رکوردهای حذف شدنی، بلافاصله بطور فیزیکی محو نمی شوند، بلکه نشانگر حذف شده می خورند تا در سازماندهی مجدد، واقعاً حذف گردند.

هم فایل داده یی و هم فایل شاخص، بلاک بندی شده اند (طول هر یک بلاک B بایت).

متوسط اندازه رکورد:

برای محاسبه متوسط اندازه رکورد، باید عوامل زیر را در نظر گرفت:

۱) حافظه لازم برای یک رکورد از ناحیه اصلی.

۲) حافظه مصرف شده برای ناحیه سرریزی به ازاء یک رکورد از ناحیه اصلی.

۳) حافظه مصرف شده برای شاخص به ازاء یک رکورد از ناحیه اصلی.

که در آن S_{Ii} ، حافظه مصرف شده برای ساختار شاخص در سطح i ام است.

برای محاسبه تعداد بلاکها را رد سطح i محاسبه و در طول بلاک ضرب می کنیم:

و چون شاخص غیر متراکم است، پس:

که در آن ei، تعداد مدخلها در سطح iام است.

و یا:

پس:

زمان واکشی یک رکورد

برای واکشی یک رکورد دلخواه، عملیات زیر باید صورت پذیرد:

۱- بررسی سر شاخص (که در حافظه اصلی است).

۲- بررسی شاخص تا رسیدن به مدخل مربوطه در سطح اول: برای این منظور، یک بلاک شاخص در هر سطح باید خوانده شود.

۳- با توجه به اینکه در عمل ۲، آدرس بلاکی که باید از ناحیه اصلی خوانده شود، به دست می آید، خواندن بلاک از ناحیه اصلی لازم است (ولی ممکن است رکورد مورد نظر در آن نباشد).

٤- به احتمالی، رفتن به ناحیه سرریزی و خواندن بلاکهایی چند.

بنابراین، برای ارزیابی زمان واکشی یک رکورد، زمانهای زیر را باید در نظر گرفت.

۱) : زمان واکشی رکورد از ناحیه اصلی.

۲) : زمان یافتن اولین رکورد از ناحیه سرریزی.

۳) : زمان واکشی رکورد از زنجیره.

زمانهای دوم و سوم با یک ضریب احتمالاتی دخالت خواهند داشت.

محاسبه:

که در آن داریم:

زمان بررسی سر شاخص

زمان طی کردن x-۱ سطح شاخص و خواندن بلاکهای شاخص

خواندن بلاک در ناحیه اصلی

با فرض به اینکه بلاکهای شاخص و بلاک داده یی در یک استوانه جای دارند، خواهیم داشت:

محاسبه : اولین رکورد از زنجیره در ناحیه سرریزی در بلاکی است که باید خوانده شود و آدرس آن از بلاک خوانده شده از ناحیه اصلی به دست آمده است.

که در آن pro عبارتست از احتمال اینکه رکورد مورد نظر در ناحیه سرریزی باشد:

: تعداد رکوردهای سرریزی است در لحظه واکشی.

و با توجه به یکی از مفروضات:

محاسبه: گفتیم که در بدترین حالت، هر رکورد از زنجیره سرریزی ها در یک بلاک جای دارد. اگر Lc طول زنجیره باشد به تعداد رکورد، در بدترین حالت این طول همان تعداد بلاک ها خواهد بود و چون مشخص نیست رکورد مورد نظر

کدام رکورد زنجیره است، پس بطور متوسط باید نصف رکوردهای زنجیره را خواند و خواهیم داشت:

در واقع Lc، تعداد رکوردهای سرریزی به ازاء یک بلاک از ناحیه اصلی است.
پس از جمع کردن سه زمان و ساده کردن داریم:

روند نمای واکشی رکورد در حالتی که رکوردها با تکنیک push-through درج شده اند:

زمان به دست آوردن رکورد بعدی:
در ارزیابی این زمان باید مشخص کرد رکورد بعدی کجاست. شش حالت وجود دارد و هر یک از حالات، به احتمالی ممکن است بروز کند. پس از در نظر گرفتن احتمالات و بیان آنها به کمک مقدار pro و ساده کردن خواهیم داشت:

و یا:

حالات ششگانه:
۱) رکورد فعلی در بلاکی از ناحیه اصلی است و رکورد بعدی نیز در همان بلاک و بلاک در بافر است (مثل فعلی رکوردهای 11.1، 11.2، 12.1 در شکل ٤-٢٦).
۲) رکورد فعلی آخرین رکورد بلاک است از ناحیه اصلی و رکورد بعدی در بلاک بعدی است از همان استوانه.
۳) رکورد فعلی آخرین رکورد بلاک از آخرین بلاک استوانه است و رکورد بعدی در بلاک بعدی است از استوانه دیگر.

٤) رکورد فعلی آخرین رکورد بلاک است و رکورد بعدی در بلاکی از ناحیه سرریزی است (مثل رکورد فعلی 11.3 و 12.3 در شکل ٤-٢٦)

٥) رکورد فعلی در بلاکی از ناحیه سرریزی است و رکورد بعدی هم در بلاکی از ناحیه سرریزی و از همان استوانه (مثل رکورد فعلی 110.1، 111.1 در شکل ٤-٢٦)

٦) رکورد فعلی در بلاکی از ناحیه سرریزی است و رکورد بعدی در بلاکی از ناحیه اصلی (مثل رکورد فعلی 110.2، 110.3، 111.2 در شکل ٤-٢٦)

روند نمای واکشی رکورد بعدی در صفحه بعد درج شده است.

روند نمای بدست آوردن رکورد بعدی

زمان درج

برای درج یک رکورد جدید، عملیات زیر باید انجام شود:

١) خواندن بلاکی که رکورد باید در آن درج شود.

٢) بازنویسی این بلاک.

٣) واکشی رکورد منطقاً پیشین برای تنظیم نشانه رو.

٤) بازنویسی همین رکورد.

زمان بهنگام سازی تغییر دهنده

ابتدا حالتی را در نظر می گیریم که در آن مقدار کلید در اثر بهنگام سازی عوض نشود و طول رکورد نیز تغییر نکند، در این حالت می توان رکورد را درجا بهنگام درآورد.

عملیات لازم:

١) واکشی رکورد بهنگام درآمدنی.

٢) کار در بافر برای ایجاد نسخه جدید.

٣) بازنویسی نسخه جدید.

و اگر بخواهیم رکورد را حذف کنیم، زمان برابر است با همان زمان
اما در حالت کلی، رکورد با وضعیت فعلی نشانگر حذف می خورد و رکورد بهنگام درآمده (نسخه جدید که باید در بافر ساخته شود)، درج می شود در ناحیه سرریزی و در واقع بهنگام سازی برون از جا انجام می شود.

زمان خواندن تمام فایل
۱) در حالت سریال:

۲) در حالت پی در پی:

زمان سازماندهی مجدد:

در این ساختار وقتی که ناحیه سرریزی پر شود، می توان فایل را سازماندهی مجدد کرد. یا اگر طول زنجیره ها طولانی شوند، که طبعاً در کارایی فایل تأثیر منفی دارد. در بعضی از سیستم ها، وقتی که برآورد نسبتاً دقیقی از حجم عملیات ذخیره سازی در دست باشد، سازماندهی مجدد را با تناوبی از پیش تعیین شده، انجام می دهند و گاه بر اساس میزان سرریزی ها، مثلا اگر ناحیه سرریزی به اندازه ۷۵٪ ناحیه اصلی برسد، فایل را سازماندهی مجدد می کنند. به هر حال عمل سازماندهی، باید در موقعی که حجم کاراهای مرکز کامپیوتر در حداقل است، انجام شود.

عملیات لازم:

۱) خواندن سریال فایل (تا بتوان مجدداً ناحیه اصلی جدید را به صورت ترتیبی ایجاد کرد).

۲) بازنویسی نسخه جدید فایل (با حذف واقعی رکوردهای حذف شدنی).

۳) بازسازی ساختار شاخص.

: بازنویسی شاخص جدید

در پایان، معایب عمده این ساختار را یادآور می شویم:

۱) عدم تقارن.

۲) ایستا بودن شاخص.

۳) مسئله درج سرریزی ها.

فایل چند شاخصی (و غیر ترتیبی) Multi-Indexed File:

مقدمه و معرفی ساختار:

این ساختار برای رفع معایبی که در ساختار ترتیبی شاخص دار دیده شد، طراحی شده است به عبارت دیگر، ساختار چنان است که پدیده عدم تقارن در آن وجود ندارد، زیرا روی تعدادی، حتی تمام صفات خاصه، می توان شاخص داشت. مسئله رکوردهای سرریزی، به صورتی که در ساختار سوم مطرح بود، در اینجا وجود ندارد، یعنی درج رکوردهای جدید آسانتر و پویاتر است و بالاخره خود ساختار شاخص وضعیتی پویا دارد و هم روند با تغییرات در فایل داده یی، قابل تنظیم و بهنگام درآوردن است.

عناصر اصلی ساختار

۱) فایل داده یی، که در بی نظم ترین شکلش می تواند حتی پایل باشد و ما در شرح این ساختار فرض می کنیم که چنین است. چنانچه هرگونه بهبود ساختاری در فایل داده یی، نسبت به پایل، در نظر گرفته شود، حالت خاصی از پایل تلقی می گردد.

۲) چندین فایل شاخص، به تعداد صفات خاصه می توان فایل شاخص داشت و بدین ترتیب کاربر می تواند، هر یک از صفات خاصه را به عنوان آرگومان

جستجو در پرس و جوهایش به کار ببرد و از سیستم درخواست کند تا شاخص روی آن صفت خاصه را ایجاد نماید.

اگر a تعداد صفات خاصه در فایل باشد، حداکثر a فایل شاخص می توان داشت.

از آنجا که در یک رکورد بطور متوسط، $\bar{a}$ تا صفت خاصه وجود دارد، لذا به یک رکورد، $\bar{a}$ ساختار شاخص ناظر است. پس این ساختار در اساس از نظر فایل داده یی، همان پایل است، اما مجهز به یک استراتژی دستیابی قوی، پویا و سریع. کاربر می تواند برای هر تعداد از صفات خاصه ای که در فایل دارد، درخواست ایجاد شاخص کند و برای واکشی سریع تک رکوردها، الزامی ندارد که حتماً از کلید اصلی به عنوان آرگومان جستجو استفاده نماید. شکل ٤-٢٧ نشان دهنده وضع کلی این ساختار است.

چون فایل داده یی پایل است، لذا درج رکوردهای جدید، مثل درج در پایل، انجام می شود، و البته ساختارهای شاخص باید متناسباً تنظیم شوند تا با تأمین پویایی، شاخص ها پیوسته بهنگام درآمده باشند.

تعریف فایل وارون (Inverted File)

وقتی که روی تمام صفات خاصه شاخص داشته باشیم، اصطلاحاً فایل را وارون می گویند. در برخی از منابع، این ساختار را، با تفاوتهایی، به عنوان ساختار وارون معرفی کرده اند.

نشست رکوردها روی رسانه:

چون فایل داده یی نظم ندارد، نشست رکوردها روی رسانه محدودیت ساختاری ندارد. البته ضوابط دیگری ممکن است مطرح باشد، از قبیل بسامد دستیابی رکوردها و یا ضوابط خاص مدیریتی، از جمله برای تأمین ایمنی بیشتر برای برخی از رکوردها، می توان آنها را به صورت پراکنده و جای جای روی رسانه درج کرد.

ضابطه انتخاب صفات خاصه شاخص:

گفتیم که حداکثر، a ساختار شاخص می توان داشت، لذا لزومی ندارد که روی تمام صفات خاصه، شاخص ایجاد شود. می توان بین صفات خاصه قائل به اولویت شد و آن صفاتی را برگزید که در بیشترین درخواست ها، به عنوان آرگومان جستجو به کار برده می شود. در واقع هدف اصلی، کاراتر شدن فایل است در پاسخگویی به نیازهای کاربر، هر چه تعداد صفات خاصه شاخص بیشتر باشد، فایل کاراتر خواهد بود و عدم تقارن آن کمتر.

اهمیت ساختار درونی فایل شاخص:

از آنجا که فایل داده یی را پایل در نظر گرفتیم و پایل فاقد هرگونه استراتژی دستیابی است، لذا آنچه در این ساختار حائز اهمیت است، همان فایلهای شاخص اند که تأمین کننده استراتژی دستیابی برای فایل داده یی می باشند. ساختار درونی فایلهای شاخص باید چنان انتخاب شود که رفتاری پویا و انعطاف پذیر داشته باشد و عملیات روی فایل را تسریع و تسهیل کند. رایجترین ساختاری که برای این منظور انتخاب می شود، ساختار B-TREE است که گونه هایی چند دارد از جمله B^+-TREE و در بحث های آتی خواهیم دید.

مثال مقدماتی برای درک بهتر ساختار:

فایلی را در نظر می گیریم با شش رکورد، حاوی اطلاعاتی در مورد موجودیت دانشجو رکوردها طول متغیر دارند و نامرتب هستند، به صورتی که در ساختار پایل دیدیم، ولی برای رعایت سادگی، ترتیب فیلدها را در نمونه های مختلف رکورد یکسان در نظر می گیریم، که تأثیری در درک نکات موجود در مثال و فهم ساختار ندارد. ضمناً در این مثال، هیچگونه ملاحظات خاص طراحی فیلدها، از قبیل ادغام کد رشته و سال ورود دانشجو در صفت خاصه شماره دانشجو، اعمال نمی کنیم. روی تمام صفات خاصه شاخص ایجاد می کنیم.

فرض می کنیم که تنها یک سطح شاخص داریم. در این ساختار شاخص متراکم است زیرا رکوردهای فایل داده یی نظم ندارند و نمی توان آنها را گروه بندی کرد.

تعداد مدخلها در سطح اول شاخص:

اگر فرض کنیم که ساختار شاخص در چند سطح ایجاد می شود، در سطح اول شاخص، بطور متوسط، برای n' رکورد، نشانه رو داریم. به عبارت دیگر n' متوسط تعداد رکوردهایی است که یک ساختار شاخص در سطح اول به آنها نشانه می رود، که همان متوسط تعداد مدخلها در سطح اول یک ساختار شاخص است، می توان رابطه زیر را بین n و n' برنهاد:

که n تعداد رکوردهای فایل داده یی است. سه حالت ممکن است وجود داشته باشد:

۱) $a'=a$ یعنی تمام صفات خاصه در تمام نمونه رکوردها موجود باشد و روی هر صفت خاصه در هر رکورد، یک مقدار مشخص داشته باشیم.

در این صورت : $n'=n$

۲) $a'<a$ در بعضی از نمونه رکوردها، بعضی از صفات خاصه را نداریم، در این صورت $n'<n$.

۳) $a'>a$ این حالت هنگامی بروز می کند که پدیده گروه اطلاع تکرار شونده و یا فقره اطلاع تکرار شونده داشته باشیم، به عبارت دیگر به ازاء یک صفت خاصه، چندین مقدار، هر یک در یک فیلد، در نمونه هایی از رکوردها موجود باشد. در این صورت $n'>n$. ما قبلا این پدیده را مورد بررسی قرار داده ایم و در اینجا برای یادآوری، دو مثال دیگر قید می کنیم:

مثال ۱: رکورد موجودیت جنس را در نظر می گیریم، قالب آن در شکل ٤-۲۸-الف دیده می شود.

یک نمونه از این رکورد چنین است:

در این نمونه رکورد گروه اطلاع قیمت سه بار تکرار شده است. اگر این اطلاع تکرار شونده را، در قالب رکورد به صورت خطی ذخیره سازی کنیم، در این صورت، برای هر مقدار از صفت خاصه قیمت، یک فیلد داریم و a<a خواهد شد.

مثال ۲: فایلی حاوی اطلاعاتی در مورد کارمندان داریم، با ۲۰/۰۰۰ رکود. در این فایل سوابق شغلی کارمندان را ذخیره کرده ایم. یک کارمند بطور متوسط ۲/۵ شغل در سابقه شغلی اش دارد. روی صفت خاصه شغل شاخص ایجاد می کنیم. این صفت خاصه تکرار شونده است.

ساختار شاخص:

دیدیم که تنها استراتژی دستیابی در این فایل، همان شاخص است. یعنی تعدادی فایل شاخص داریم ناظر بر رکوردها. فایل شاخص باید ساختاری داشته باشد که بتوان آن را، همروند با عملیات روی فایل داده یی، بطور پویا بهنگام درآورد. راه حل مطروحه در فایل ترتیبی شاخص دار که ایجاد شاخص ایستا و زنجیره رکوردهای سرریزی بود. در اینجا کارایی ندارد، زیرا برای یک رکورد واحد، ممکن است چندین زنجیره وجود داشته باشد و مدیریت این زنجیره ها دشوار و زمانگیر خواهد بود، ضمن اینکه دستیابی ساختار شاخص هم عیب عمده ای به شمار می رود.

راه حل دیگر عبارتست از در نظر گرفتن یک ساختار شاخص تک سطحی برای هر صفت خاصه شاخص. چنین ساختاری نیز فاقد سرعت کافی خواهد بود.

راه حل اساسی این است که در ایجاد فایل شاخص از یک مدل تئوریک ساختمان داده ها استفاده کنیم که رفتاری پویا داشته باشد. مناسبترین مدل همان B-

TREE است. یادآور می شویم که B-TREE درختواره ای است که در آن عمق تمام شاخه ها، از ریشه تا گره های انتهایی یکسان است. شکل ٤-٢٩ وضع کلی یک درختواره شاخص را نشان می دهد.

مثال:

نمونه ای از یک درختواره شاخص و فایل داده یی که فاقد نظم است. این مثال، سادگی خاص دارد و جنبه های ویژه ساختار شاخص در آن مشهود نیست.

پایین ترین سطح شاخص، دارای نظم است و در بعضی از سیستم ها (مثلا VSAM) به آن مجموعه توالی (sequence set) می گویند.

نحوه پیاده سازی فایل شاخص با استفاده از B-TREE:

هر بلاک شاخص (یا تعدادی بلاک با نامی دیگر) گرهی از درختواره در نظر گرفته می شود. یک گره، در لود اولیه، به تمامی پر نمی شود، به عبارت دیگر چگالی آن در لود اولیه کوچکتر از یک است. ساختمان درونی یک گره (با فرض اینکه هر گره یک بلاک باشد) به صورتی است که در شکل ٤-٣١ دیده می شود.

اگر طول هر مدخل V+P بایت باشد، y اسمی، برابر است با:

این ظرفیت نشانه روی، در لود اولیه به تمامی پر نمی شود، بلکه تعدادی مدخل به عنوان رزرو منظور می گردد. ظرفیت واقعی نشانه روی در لود اولیه (effective fanout) که آن را به y_{eff} نشان می دهیم حداکثر برابر با y و حداقل است.

یعنی در لود اولیه، اقلا نیمی از مدخلهای یک گره درختواره پر است. معمولاً $y_{eff}=0/69y$ در نظر گرفته می شود (پژوهشگران نشان داده اند که این ضریب، مناسبترین است). حال ببینیم که درختواره شاخص در عملیات روی فایل، چگوهه متناسبا تنظیم می گردد.

در درج:

برای درج رکورد آن را در فایل داده یی (که فرض کردیم پایل است) اضافه می کنیم. پس از درج، باید ارتباط ساختاری رکورد بلافاصله برقرار شود، یعنی مدخل شاخص مربوطه در سطح اول شاخص ایجاد گردد. این مدخل طبعاً باید در بلاک شاخص مربوطه (با توجه مقدار صفت خاصه شاخص و محل منطقی آن در مجموعه توالی) به وجود آید.

چون تعدادی مدخل رزرو در هر بلاک شاخص داریم، لذا ایجاد مدخل شاخص ناظر به رکورد درج شده به آسانی انجام می پذیرد (آدرس محل نشست در فیلد نشانه رو گذاشته می شود). تا زمانیکه مدخل خالی در بلاک شاخص موجود باشد، تنظیم درختواره با ایجاد مدخل جدید، پایان می پذیرد و عمل دیگری لازم نیست. به عبارت دیگر، تغییر عمده ای در درختواره پدید نمی آید. اما اگر برای ایجاد مدخل در بلاک شاخص مربوطه در سطح اول دیگر جا نباشد یعنی yeff برابر با y شده باشد، در این صورت بلاک شاخص پر شده را باید تقسیم کرد (اصطلاحا عمل spliting باید انجام شود).

نحوه عمل چنین است:

یک بلاک شاخص خالی، موسوم به بلاک همراه (partner) به فایل شاخص اختصاص داده می شود و نیمی از مدخلهای بلاک پر شده، با رعایت ترتیب، به این بلاک انتقال می یابند. این بلاک همیشه در سمت راست بلاک پر شده ایجاد می شود و بدینسان گرهی جدید، در همان سطح گره پر شده، در درختواره پدید می آید. این گره جدید طبعا باید با گرهی در سطح بلافاصله بالاتر مرتبط شود، به عبارت دیگر در گرهی از سطح بالاتر باید مدخلی جدید ایجاد شود تا به بلاک همراه نشانه رود. پس در عمل تقسیم بلاک پر شده، اقلا سه بلاک شاخص باید ایجاد و یا بهنگام درآیند. اگر بلاک سطح بالاتر هم پر بوده باشد، خود نیاز به تقسیم شدن خواهد داشت و بلاکی همراه باید برای آن ایجاد شود. بدین ترتیب

عمل تقسیم بلاکهای شاخص، در یک شاخه از درختواره، ممکن است حتی تا ریشه شیوع یابد و در این صورت، عمق درختواره از x به x+1 تغییر می کند. در شکل ٤-٣٢ مثالی می بینیم که نحوه عمل را نشان می دهد.

در این مثال، یک درختواره شاخص، درست پس از لود اولیه با $y_{eff}=0/75y$ نشان داده شده است. پس از درج R_6 خواهیم داشت:

حال اگر رکورد R7 را درج کنیم، بلاک دیگر جا ندارد، لذا باید، به صورت زیر، تقسیم شود:

باید مدخلی در بلاک سطح بالاتر ایجاد شود مطابق شکل زیر:

مثالی دیگر:

این مثال نحوه ایجاد مدخل شاخص را در عمل درج، در VSAM نشان می دهد. ما ویژگیهای این سیستم را در فصل مربوطه خواهیم دید. در اینجا توضیح کوتاهی در مورد VSAM ارائه می کنیم:

در این سیستم، پس از ایجاد مجموعه توالی، هر مدخل از یک بلاک شاخص نشانه می رود به گروهی از رکوردها موسوم به فصاله کنترل (CA) و تعدادی CA تشکیل یک ناحیه کنترل (CI) را می دهند. برای درج رکوردها، تا زمانی که فاصله کنترل آزاد در یک ناحیه کنترل وجود داشته باشد، رکورد در آن فاصله درج و در صورت لزوم، مدخل شاخص مربوطه ایجاد می گردد و اگر دیگر فاصله کنترل آزاد وجود نداشته باشد، ناحیه کنترل به دو ناحیه تقسیم می شود و عمل تقسیم مدخلهای شاخص در مجموعه توالی نیز انجام می پذیرد. توجه داریم که

در این مثال، ضمناً نحوه تقسیم ناحیه داده یی نیز دیده می شود و البته این جنبه VSAM در ساختار چهارم مورد نظر نیست.

در حذف:

رکورد حذف شدنی از فایل داده یی حذف می شود (با ضبط نشانگر حذف شده در آن) و در پی آن، ارتباط ساختاری رکورد با استراتژی دستیابی، یعنی درختواره شاخص باید از بین برود. برای این منظور، مدخل شاخص ناظر به رکورد در یک بلاک مربوطه از سطح اول شاخص باید اصلاح شود، یعنی آدرس رکورد حذف شده، موجود در مدخل باید محو گردد.