Sıralı Erişimli Dosyalar

Dosya

Fiziksel depolama ortamlarında verilerin saklandığı mantıksal yapılara dosya denir.

- Dosyalar iki şekilde görülebilir.
  - **Byte dizisi şeklinde** veya
  - Alanlar içeren kayıtların bir arada bulunması şeklinde

```
87358CARROLL ALICE IN WONDERLAND
03818FOLK FILE STRUCTURES
79733KNUTH THE ART OF COMPUTER PROGRAMMING
86683KNUTH SURREAL NUMBERS
18395TOLKIEN THE HOBITT
```

Dosyaların byte dizisi şeklindeki yapısı

Bu yaklaşımda verilerin anlamı kaybolur ve bunları geriye getirmek için (yeniden ayırmak için) yol yoktur.

Alan, Kayıt Organizasyonu

Kayıt (Record): İlişkili alanların biraraya gelmesiyle oluşan yapıdır.

Alan (Field): Bir dosya içindeki mantıksal anlamı olan en küçük birimdir.

Anahtar (Key): Bir kayıt içinde bulunan ve kaydı tekil (eşsiz) olarak tanımlayan, belirten alanlar kümesidir.

Örn: T.C kimlik no, ISBN no,tel no vb...

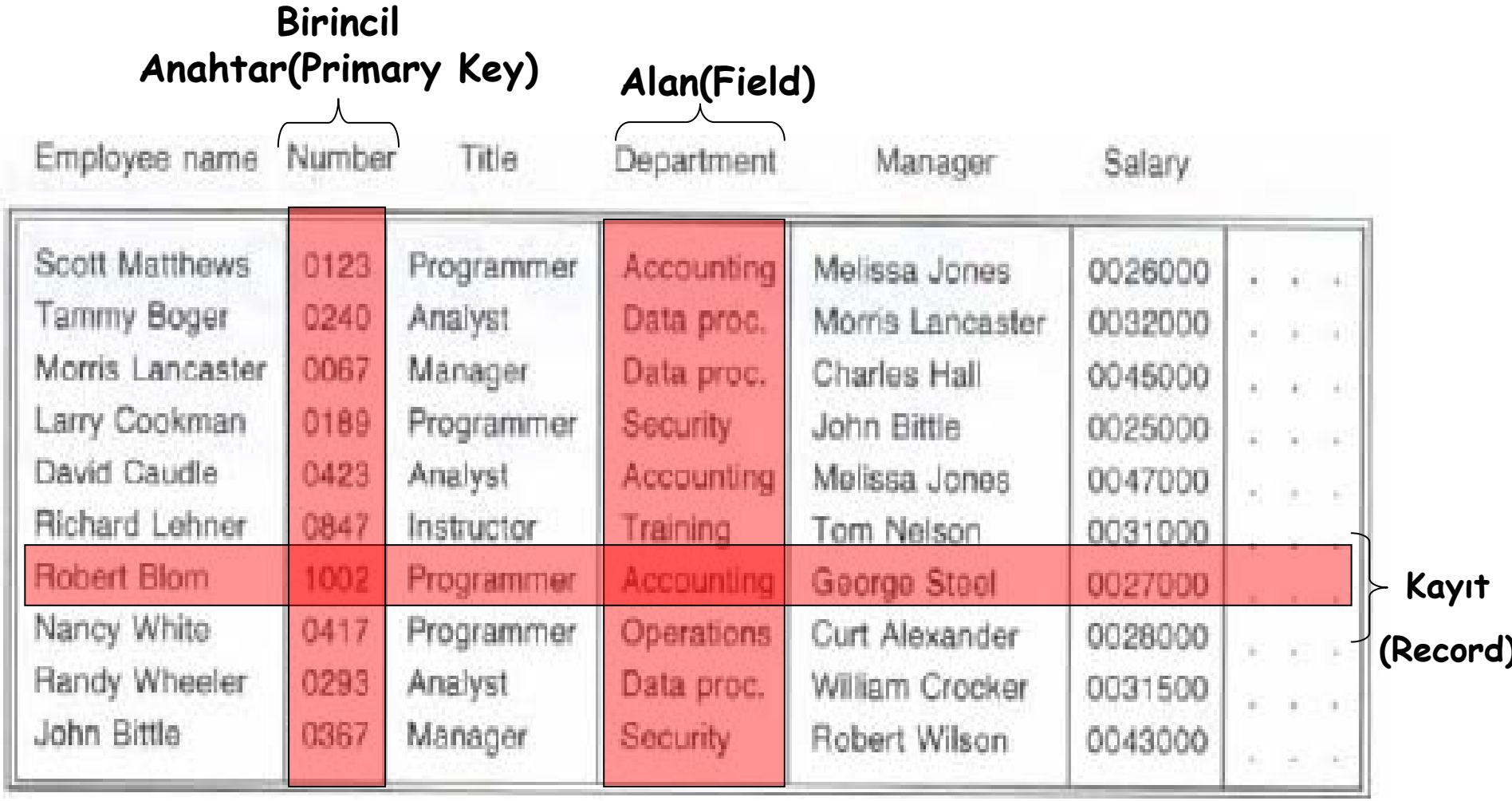
Anahtarlar

Birincil Anahtar (Primary Key): Bir kaydı tekil olarak tanımlayan anahtara birincil anahtar denir.

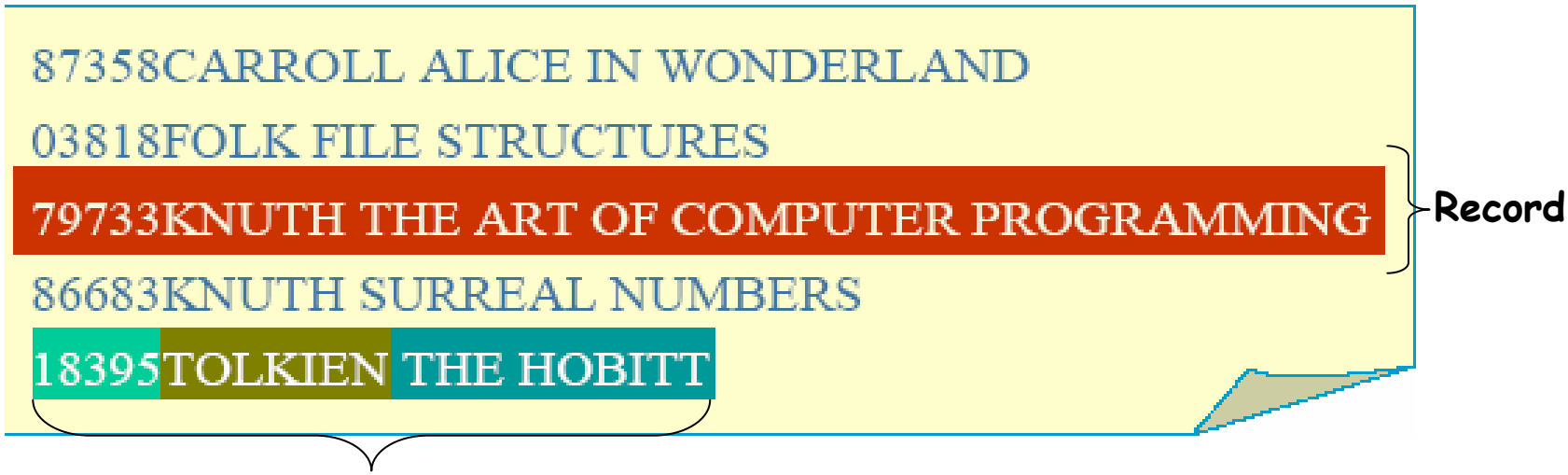
İkincil Anahtar (Secondary Key): Arama için kulanılabilecek diğer anahtarlardır.

Örn: Yazar adı, kitap adı veya yazar adı + kitap adı

Anahtarlar alanlara karşılık gelir, bazı anahtarlar özel olarak dikkate alınırlar, bazen ise birden fazla alanın kombinasyonu da anahtar gibi düşünülüp aramalarda kullanılabilir.


Birincil Anahtar(Primary Key)
Alan(Field)
Employee name
Number
Title
Department
Manager
Salary
Scott Matthews 0123 Programmer Accounting Melissa Jones 0026000
Tammy Boger 0240 Analyst Data proc. Morris Lancaster 0032000
Morris Lancaster 0067 Manager Data proc. Charles Hall 0045000
Larry Cookman 0189 Programmer Security John Bittle 0025000
David Caudle 0423 Analyst Accounting Melissa Jones 0047000
Richard Lehner 0847 Instructor Training Tom Nelson 0031000
Robert Blom 1002 Programmer Accounting George Stool 0027000
Nancy White 0417 Programmer Operations Curt Alexander 0028000
Randy Wheeler 0293 Analyst Data proc. William Crocker 0031500
John Bittle 0367 Manager Security Robert Wilson 0043000
Kayıt (Record)

deneme.txt



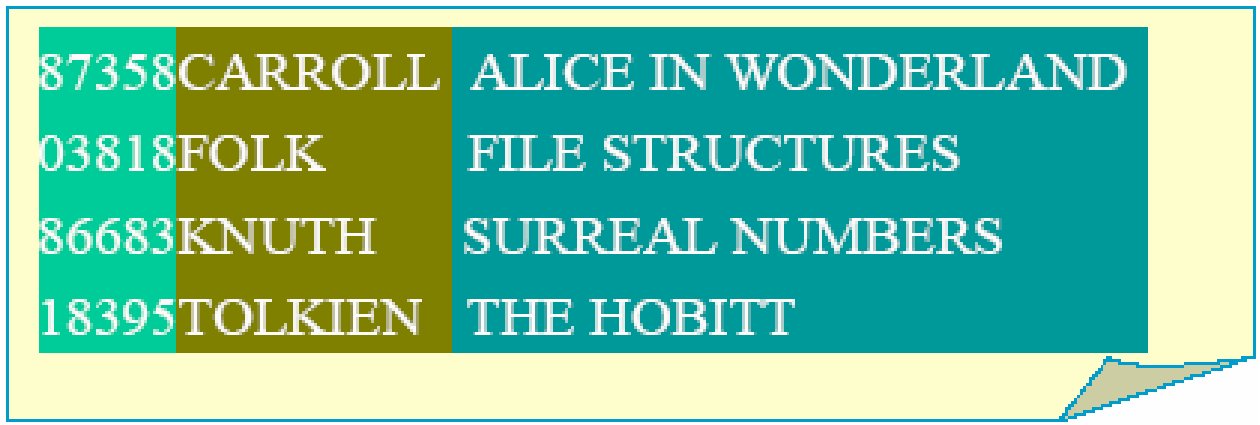
ISBN numarası, yazar adı ve eser adını içeren 3 farklı alan.

Alan Yapıları

Alan yapılarını 5 kategoride incelemek mümkündür.

- Sabit Uzunluklu Alanlar (Fixed Length)
- Uzun Göstergesi ile Başlayan Alanlar (Begin Each Field with Its Length Indicator)
- Her Birinin Sonunda Sınırlayıcıların Bulunduğu Yapılar (Delimiters to Seperate Fields)
- Anahtar Kelime=Değeri (Keyword=Content) Şeklinde Depolanan Alanlar

Sabit Uzunluklu Alanlar (Fixed Length Fields)



Uzunluk Göstergesi ile Başlayan Alanlar (Begin Each Field with Its Length Indicator)

05873580 7CARROLL19ALICE IN WONDERLAND
05038180 4FOLK15FILE STRUCTURES
05866830 5KNUTH15SURREAL NUMBERS
05183950 7TOLKIEN10THE HOBITT

Her Birinin Sonunda Sınırlayıcıların Bulunduğu Yapılar (Delimiters to Seperate Fields)

87358|CARROLL|ALICE IN WONDERLAND|
03818|FOLK|FILE STRUCTURES|
86683|KNUTH|SURREAL NUMBERS|
18395|TOLKIEN|THE HOBITT|

Anahtar=Kelime Değeri (Keyword=Contet) Şeklinde Depolanan Yapılar

ISBN=87358|**AU**=CARROLL|**TI**=ALICE IN WONDERLAND|
ISBN=03818|**AU**=FOLK|**TI**=FILE STRUCTURES|
ISBN=86683|**AU**=KNUTH|**TI**=SURREAL NUMBERS|
ISBN=18395|**AU**=TOLKIEN|**TI**=THE HOBITT|

Özet

TÜR	Avantaj	Dezavantaj
Sabit Uzunluklu	Okuma/Yazma Kolay	Boşluklarla doldurulan ve işe yaramayan alan
Uzunluk Göstergeli	Bir sonraki alana sıçramak kolay	Büyük alanlar, uzunluk bilgisi için 1 byte'tan fazla yer gerektirir.
Sınırlayıcılı	Uzunluk göstergeliden daha az boş alan kullanılır	Alandaki tüm karakterlerin teker teker kontrol edilmesi gerekir.
Anahtar Kelimeli	Alanlar kendilerini açıklarlar, kayıtta eksik alanlar olabilir.	Anahtar kelimelerle boşa alan kaybedilir.

Kayıt Yapıları

- Sabit Uzunluklu Kayıtlar
- Sabit Sayıda Yer İçeren Kayıtlar
- Uzunluk Göstergesi ile Başlayan Kayıtlar
- Adresleri İzleyen Bir Indexin Kullanımı
- Sonunda Sonlandırıcı(Sınırlayıcı)'nın Bulunduğu Kayıtlar

Sabit Uzunluklu Kayıtlar

İki farklı şekilde yapılabilir.

i) Sabit Uzunluklu Alanlar İçeren Sabit Uzunluklu Kayıt

203572	Alan Tharp File Organizaion
062833	John White File Organization

ii) Değişken Uzunluklu Alanlar İçeren Sabit Uzunluklu Kayıt

203572	Alan Tharp File Organization	
062833	Micheal John White File Organization	

Kullanılmayan Alan

Değişken Uzunluklu Kayıtlar

i) Sabit Sayıda Alan İçeren Kayıtlar

203572	Alan Tharp File Organizaion	062833	John White File Organization

ii) Uzunluk Göstergesi ile Başlayan Kayıtlar

33203572	Alan Tharp File Organizaion	**35** 062833	John Fisher File Organization

iii) Adresleri İzleyen Bir Indexin Kullanımı

Index dosyası her bir kaydın dosya içindeki başlangıç adresini tutar. Kayıtların başlangıcını bulmak için bu index üzerinde arama yapılır. Genellikle index sabit uzunlukta olur.

Değişken Uzunluklu Kayıtlar (Devam)

iv) Sonunda Sonlandırıcı Bulunan Kayıtlar

Bu yapıda kayıdın sonunda “end of line” gibi bir ifade bulunur.

Tür	Avantaj	Dezavantaj
Sabit Uzunluklu Kayıtlar	**İstenilen Kayda Kolayca Konumlanma**	**Boş Alan Kullanımı**
Değişken Uzunluklu Kayıtlar	**Boş alan daha az kullanılmakta**	**Bir index dosyasına gereksinim duyulur.**

Dosya Organizasyonu

- Sıralı Erişimli Dosyalar (Sequential)
- Doğrudan Erişimli Dosyalar (Direct)
- Index Yapılı Dosyalar (Indexed)

Dosya İşlemleri

Temel dosya işlemleri :

- Bir kaydın alınması
- Bir kaydın eklenmesi
- Bir kaydın silinmesi
- Bir kaydın içindeki alanın değiştirilmesi

Sıralı Erişimli Dosyalar (Sequential Files)

- Kayıtlar bitişik (ardışık) olarak saklanır.

1	**2**	**3**	**. . . . . .**	**i**	**i+1**	**i+2**	**. . . . . .**	**n-2**	**n-1**	**n**

Kayıtlara Erişim (Access)

- Sıralı erişim için bu dosya yapısı olukça uygundur. Bir kayıttan diğer kayda geçmek için aktif olan kaydın adresini, kayıt boyutu kadar arttırmak yeterlidir.

- Sıralı dizi yapısı bu yapıya örnek olarak verilebilir.

- Döngü sayacı kullanılarak sırasıyla kayıtlara erişerek işlem yapmak mümkündür.

- Bu durum, döngü sayacının primary key olduğunda geçerlidir. Örn: Number sütununa göre

Employee name	Number	Title	Department	Manager	Salary	
Scott Matthews	0123	Programmer	Accounting	Melissa Jones	0026000	. . .
Tammy Boger	0240	Analyst	Data proc.	Morris Lancaster	0032000	. . .
Morris Lancaster	0067	Manager	Data proc.	Charles Hall	0045000	. . .
Larry Cookman	0189	Programmer	Security	John Bittle	0025000	. . .
David Caudle	0423	Analyst	Accounting	Melissa Jones	0047000	. . .
Richard Lehner	0847	Instructor	Training	Tom Nelson	0031000	. . .
Robert Blom	1002	Programmer	Accounting	George Stool	0027000	. . .
Nancy White	0417	Programmer	Operations	Curt Alexander	0028000	. . .
Randy Wheeler	0293	Analyst	Data proc.	William Crocker	0031500	. . .
John Bittle	0367	Manager	Security	Robert Wilson	0043000	. . .

- Eğer erişim işlemi name alanına göre yapılacaksa sıralı arama (sequential search) yapılmalıdır.

- Sıralı aramada, istenilen kayda ulaşılıncaya kadar bütün kayıtlara sırasıyla teker teker bakılır.

  Örn: “Rober Blom” adlı kişiye ulaşabilmek için 7 tane probe’a gereksinim vardır.

  Probe: Farklı bir konuma ulaşıma verilen isimdir.

- Başarısız aramalarda dosyadaki tüm kayıtlara bakılması gerekir. Bu yüzden başarısız aramalarda sıralı erişimde zaman kaybına neden olur.

- Genel olarak (n bir dosyadaki toplam kayıt sayısı olsun)
  - Minimum 1 kayda erişilecek veya
  - Maksimum n tane kayda erişilecektir.

  } Ortalama= (n+1)/2 → n/2 erişim söz konusudur.

- Kayıtlar birincil anahtarlarına göre (sayısal veya alfabetik olarak) sıralanırsa, istenilen kayda ulaşmak daha az zaman alır.

- Büyük n değerleri için performans kaybı söz konusu iken, n'in az olduğu durumlar için oldukça yapısı basit olduğundan tercih edilebilir.

- Computational Complexity O(n)'dir.

Binary Search

- Kayıtlar sıralı durumdadır.
- Aranan alanın ortasından aramaya başlanır.
- Her defasında kayıtların yarısı elenir.
- Computational Complexity O ($\log_2 n$)'dir.

Binary Search

```
        proc binary_search

             /*  The n records of the file are ordered in in-
                 creasing order of the keys.   */

1            LOWER := 1
2            UPPER := n
3            while LOWER ≤ UPPER do
4                 MIDDLE := ⌊ (LOWER + UPPER) / 2 ⌋
5                 if key[sought] = key[MIDDLE]
6                      then terminate successfully.
7                      else if key[sought] > key[MIDDLE]
8                           then LOWER := MIDDLE + 1
9                           else UPPER := MIDDLE - 1
10           end
11           terminate unsuccessfully.
        end binary_search
```

Örnek

[13,	16,	18,	27,	28,	29,	38,	39,	53]
[13,	16,	18,	27],	28,	29,	38,	39,	53
13,	16,	[18,	27],	28,	29,	38,	39,	53
13,	16],	[18,	27,	28,	29,	38,	39,	53

^ → aktif olan kayıt

Interpolation Search

- Telefon rehberinde bir kişinin aranmasına benzer.

- Aranan kaydın pozisyonunun tahmin edilmesiyle bir sonraki pozisyon belirlenir.

$$NEXT := \left\lceil LOWER + \frac{key[aranan] - key\ (LOWER)}{key[UPPER] - key\ (LOWER)} (UPPER\text{-}LOWER) \right\rceil$$

- Kayıtlar birincil bellekte ise binary search, ikincil bellekte ise interpolation search tercih edilir.

Self-Organizing Sequential Search

- Bu yapıda sık kullanılan kayıtlar dosyanın başlangıcına getirilerek performans arttırılmaya çalışılır.

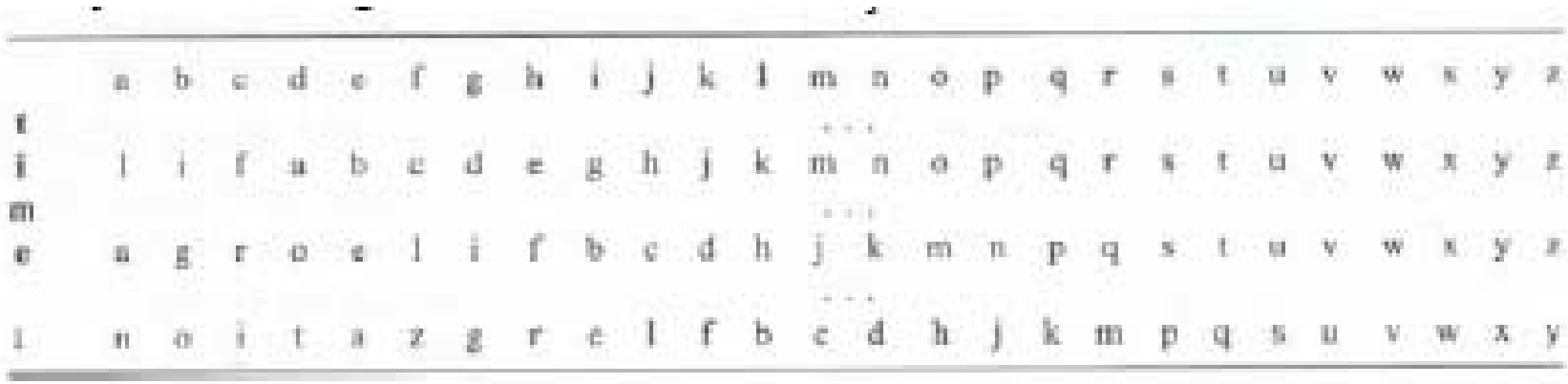
  Bunun için 3 yaygın algoritma kullanılır:

- Move_to_front
- Transpose
- Count

Move_to_Front

- Bir kayda ulaşıldığında dosyanın başına alınır.
- Bir kayıt eğer sık kullanılmıyorsa diğer kayıtların erişim süresini arttırır.
- Yer kısıtlı olmadığı ve hızlı erişimin önemli olduğu durumlarda kullanılır.
- Bağlı yapılar kullanılır.

Örnek



Transpose

- Bir kayda ulaşıldığında önündeki ile yer değiştirir.
- Bir kayıt dosyanın başlangıcına gelinceye kadar birden fazla ulaşılmalıdır.
- Move_to_front algoritmasına göre daha kararlıdır.
- Yer kısıtlı olduğunda tercih edilmelidir.

Count

- Kayıtlar erişilme sırasına göre büyükten küçüğe doğru sıralanır.
- Erişilme bilgisini tuttuğu için fazladan alan gerektirir.
- Erişim sayıları başka bir yerde gerekiyorsa bu yapı tercih edilebilir.