データベースの
アーキテクチャ

データベース論Ⅰ

第2回

URL　http://homepage3.nifty.com/suetsuguf/

作成者　　末次文雄 ©

目次

１．アーキテクチャ

１．１　アーキテクチャとは？

１．２　データベースのアーキテクチャとは？

１１．アーキテクチャとは？

①　アーキテクチャの語源

・archtecture

アーキテクチャ　建築様式

アーキテクト　　建築家

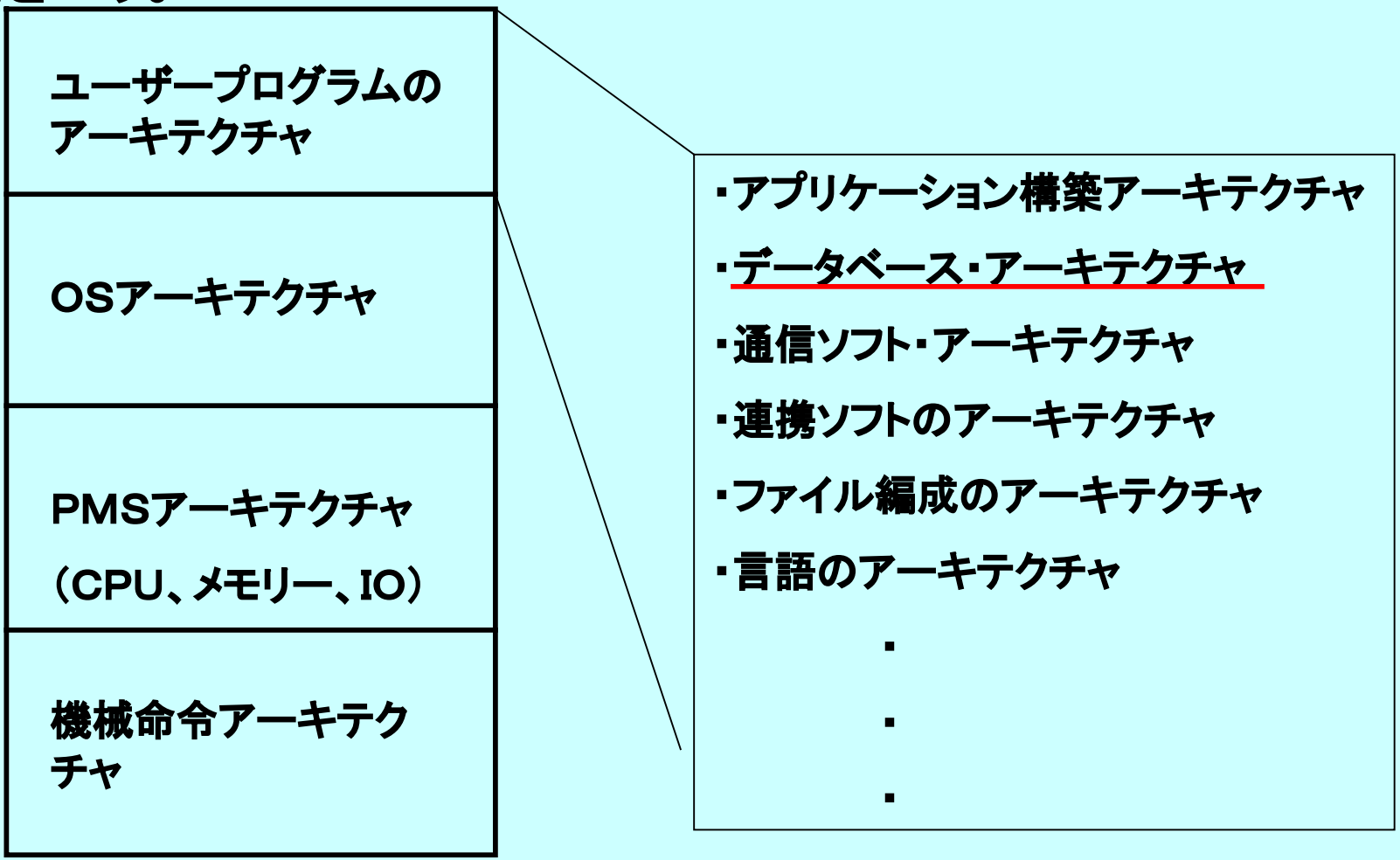
（例）Gothic　archtecture

ゴシック建築様式

② 電子計算機のアーキテクチャ

計算機のある機能レベルでの、実現方法、下位機能構成の構造をいう。

（例）



１．２　データベースのアーキテクチャとは？

- データベースの実現方法、下位構成の基本機能の構造を決めたもの。
  - －データモデリング
    ネットワーク構造、階層構造、
    リレーショナル構造、オブジェクト指向
  - －DBMSの構造
  - －ファイルの編成方式

第2回講義の対象

  - －使用する計算機の構成
    （ホスト中心、CS型、分散型）

２．データ独立性

２．１　データベース誕生以前の問題点（復習）

２．２　データ独立性の目的

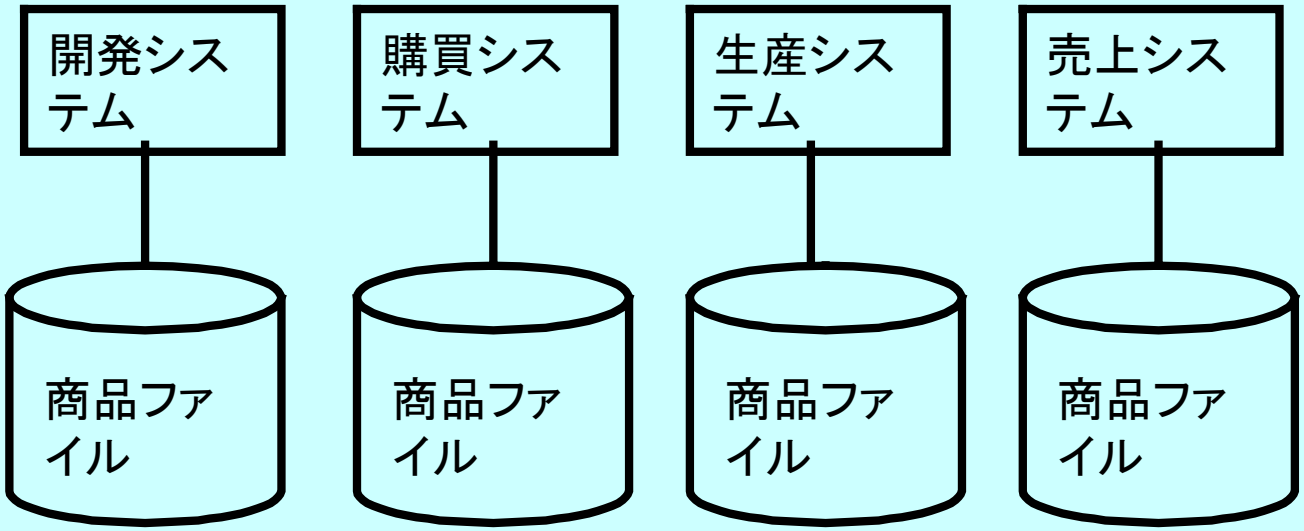
２．３　データ独立性の方法

２．４　DBMSの機能（復習）

２．１データベース誕生以前の問題点（復習）

①　データの重複

・サブシステム毎、プログラム毎に、専用のファイルを作成しており、データの重複が著しい。



②　データ重複による問題点

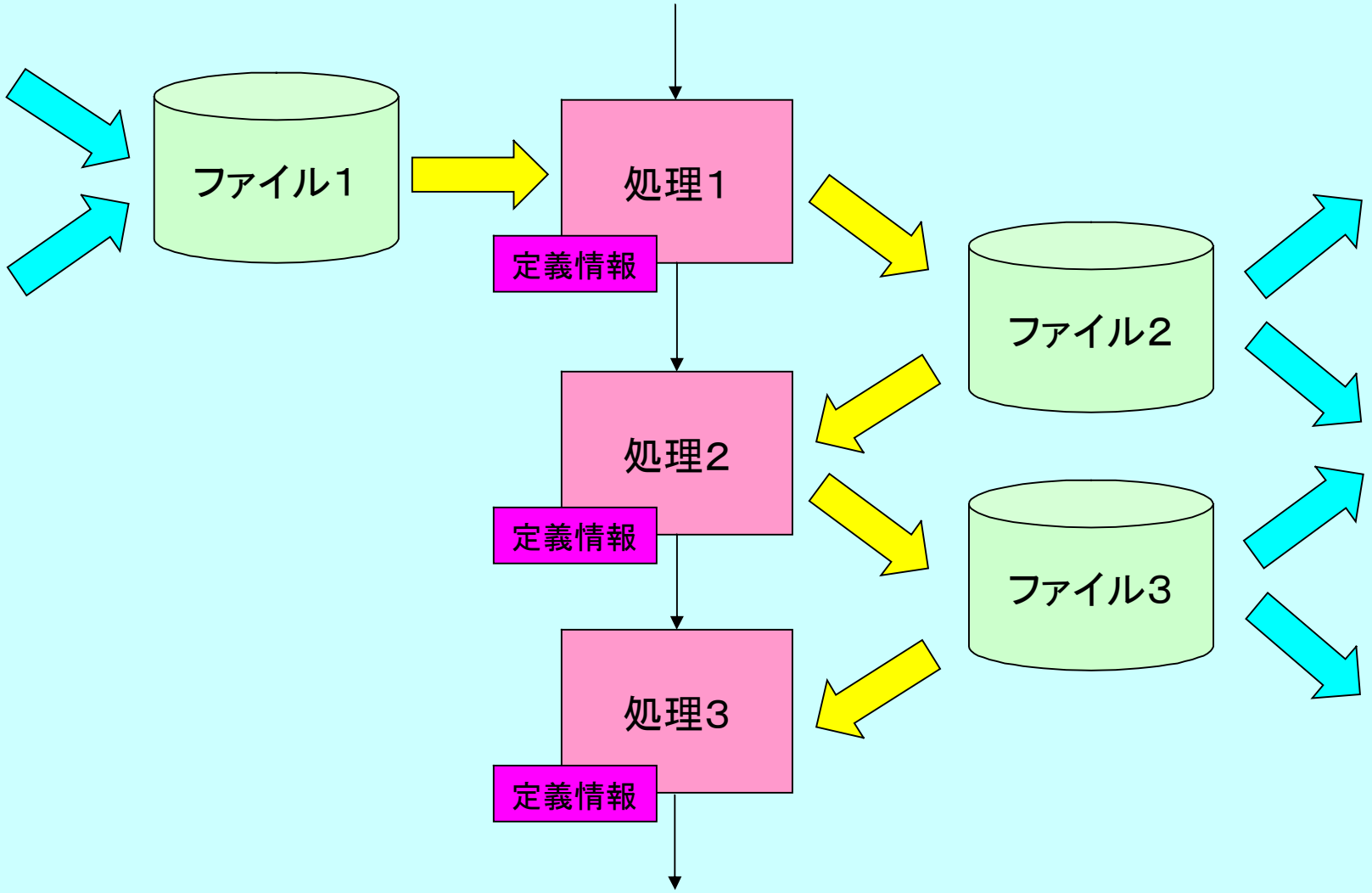
- データ量の増大→保管費用の増大
  →データ更新の工数が増大
- データの不整合→部門間で業務が繋がりにくい
- 類似データの存在→データの変換が必要
- システム変更が複雑→システム維持費用の増大
- データ取出方法のバラツキ→同上

③ データ共有化の問題

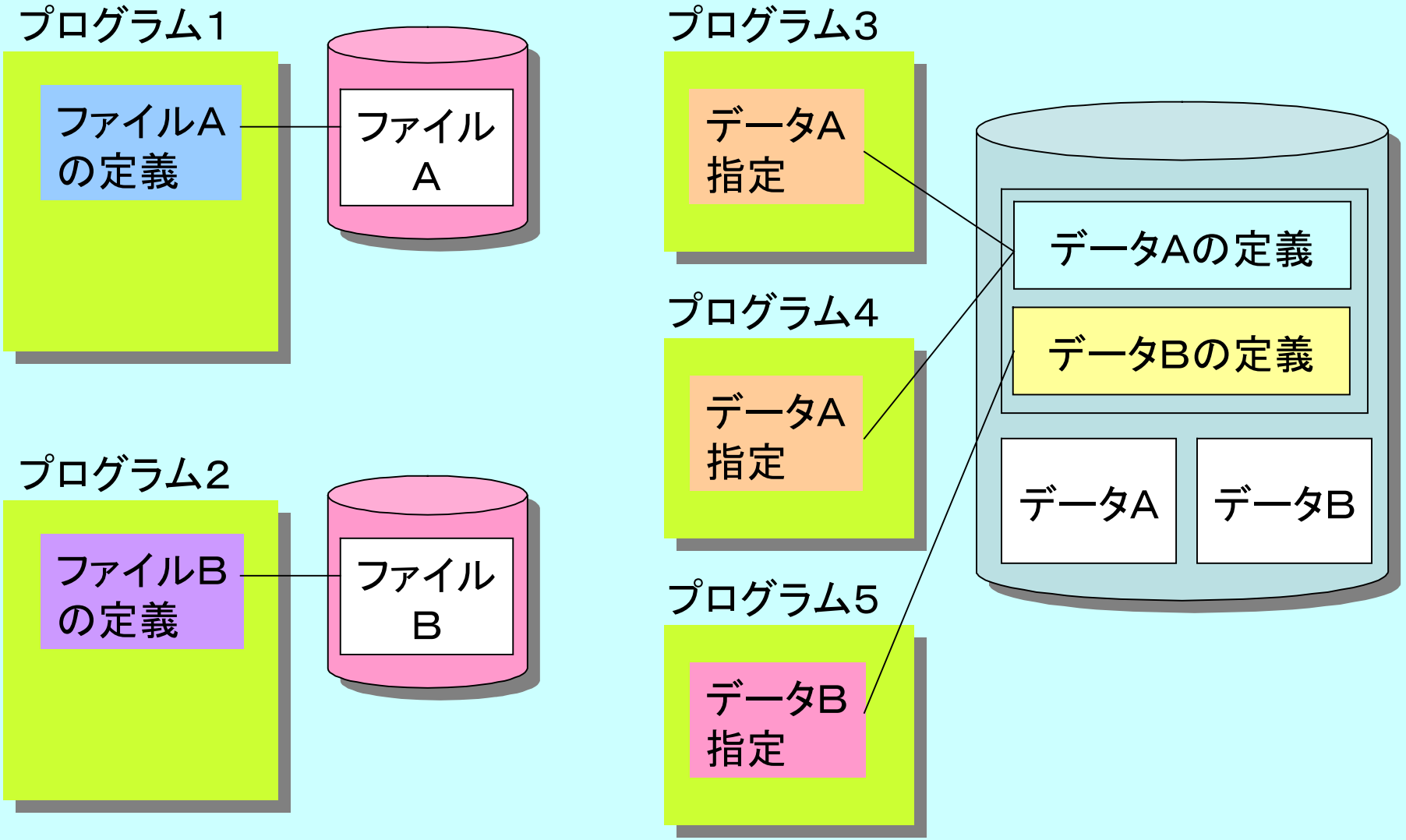
単にファイルを統合化した場合は、以下の
問題が、新たに発生する。

- 同時にファイル利用する場合に、
  ー更新前後でデータ内容が異なる
  ーそのため、更新と同時には使用出来なくなる
- ファイル障害発生時に、その影響範囲が増大する
  ー機器障害時、故意の破壊時
- 統合ファイルが不正に使用された場合、影響大
- ファイルの変更頻度が増え、それが影響する
  プログラムが増大
- ファイル使用時に専門的な知識が必要である

データベース誕生以前のデータの流れ



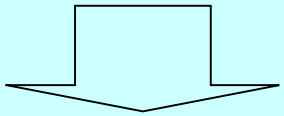
ファイル管理システムとデータベース



(1)ファイル管理システム　　(2)データベース

２．２　データ独立の目的

・ファイルのデータ構造、データ定義、
　格納場所などが変更されても、
それを使用するアプリケーション・プログラム
　に対する影響を最小限に抑える。

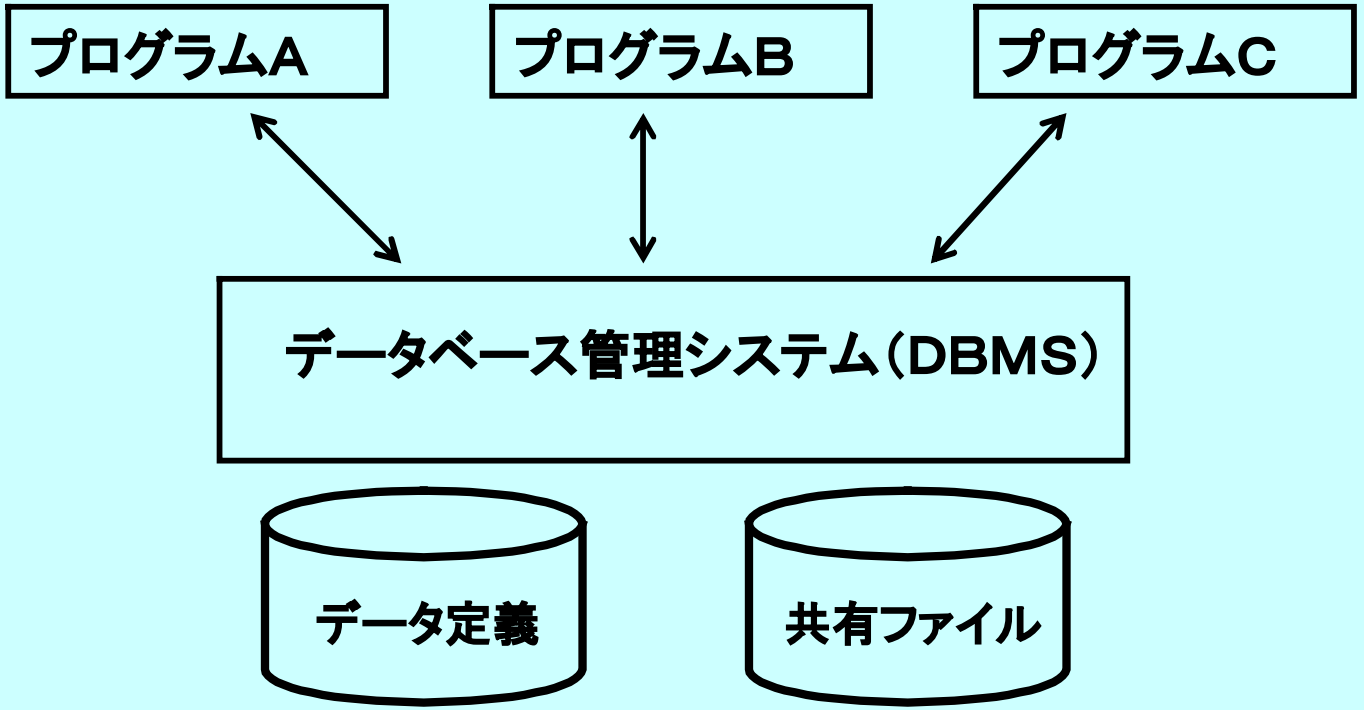


・そのためには、レコードやファイルの
　－物理的な記憶媒体、記憶場所、
　－ファイルの編成方法、
　－レコード内のフィールドの配置
　－レコードの順序
などの情報をﾌﾟﾛｸﾞﾗﾑから独立させて定義し保管する。

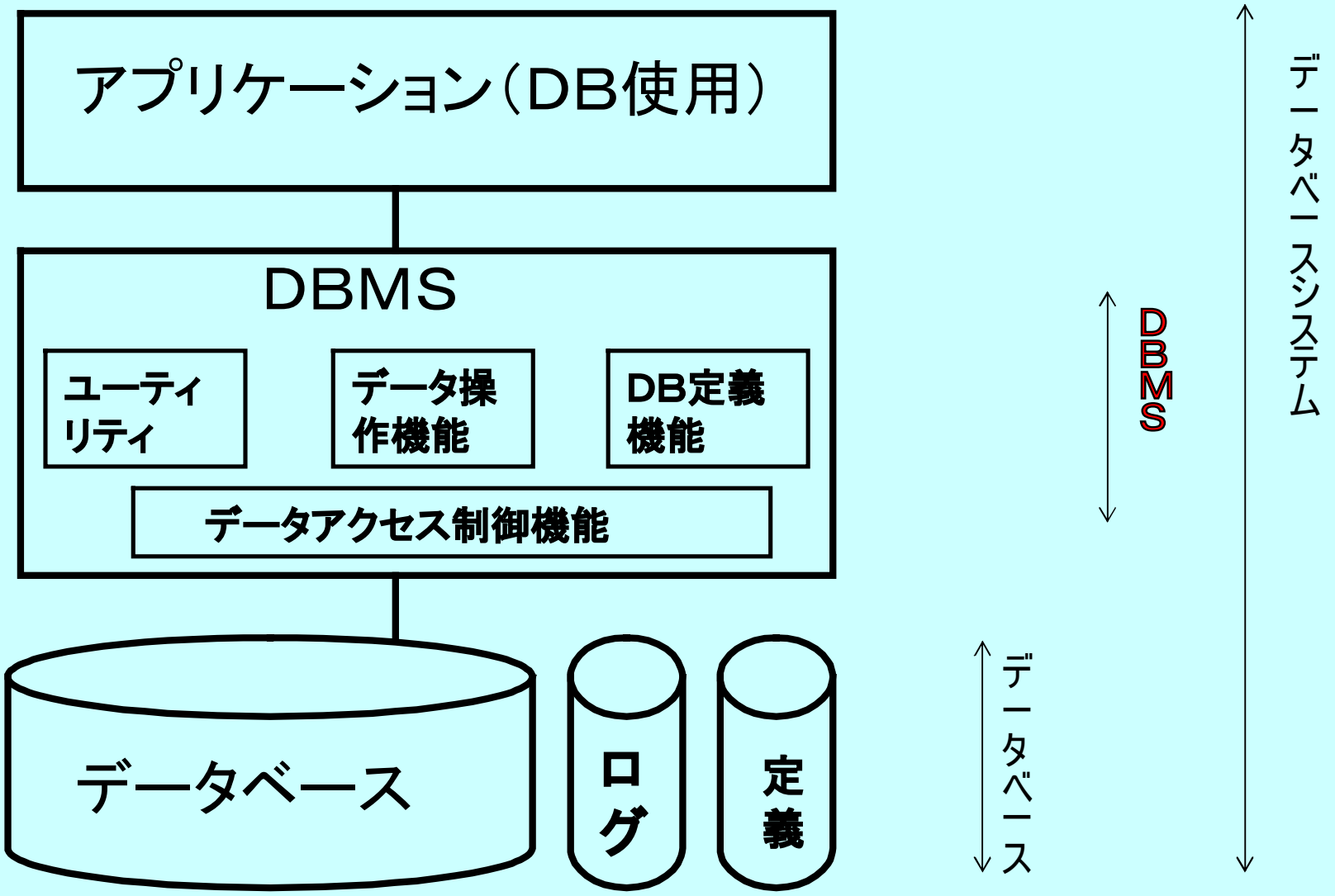
２．３　データ独立の方法

データをプログラムから独立させる方法は、

データベースとプログラムの間に緩衝地帯を設ける。



DBMSのシステム構成



２．４　DBMSの機能（復習）

データの独立性	データが個々のプログラムに従属しない
データの一貫性	同時更新、同時利用時でも、データに矛盾が生じない
容易なデータ操作	共有データをプログラムから容易に使用できる
機密保護	許可された者以外は、データ利用が出来ない
障害復旧	異常発生時でも、データを正確に元の状態に戻せる

３．データベースのスキーマ構造

３．１　スキーマとは？

３．２　3層が必要な理由

３．３　データベースの3層構造

３．４　DBMSとスキーマの関係

３．１　スキーマとは？、

- スキーマ （Schema）
  ー原義：図表、図式
- データベースのスキーマ
  ーデータベースに関する、いろいろな型を
  定義したものの集まりの意
  ーCODASYL委員会が最初に使用
  （「枠組み」の意味で使用）

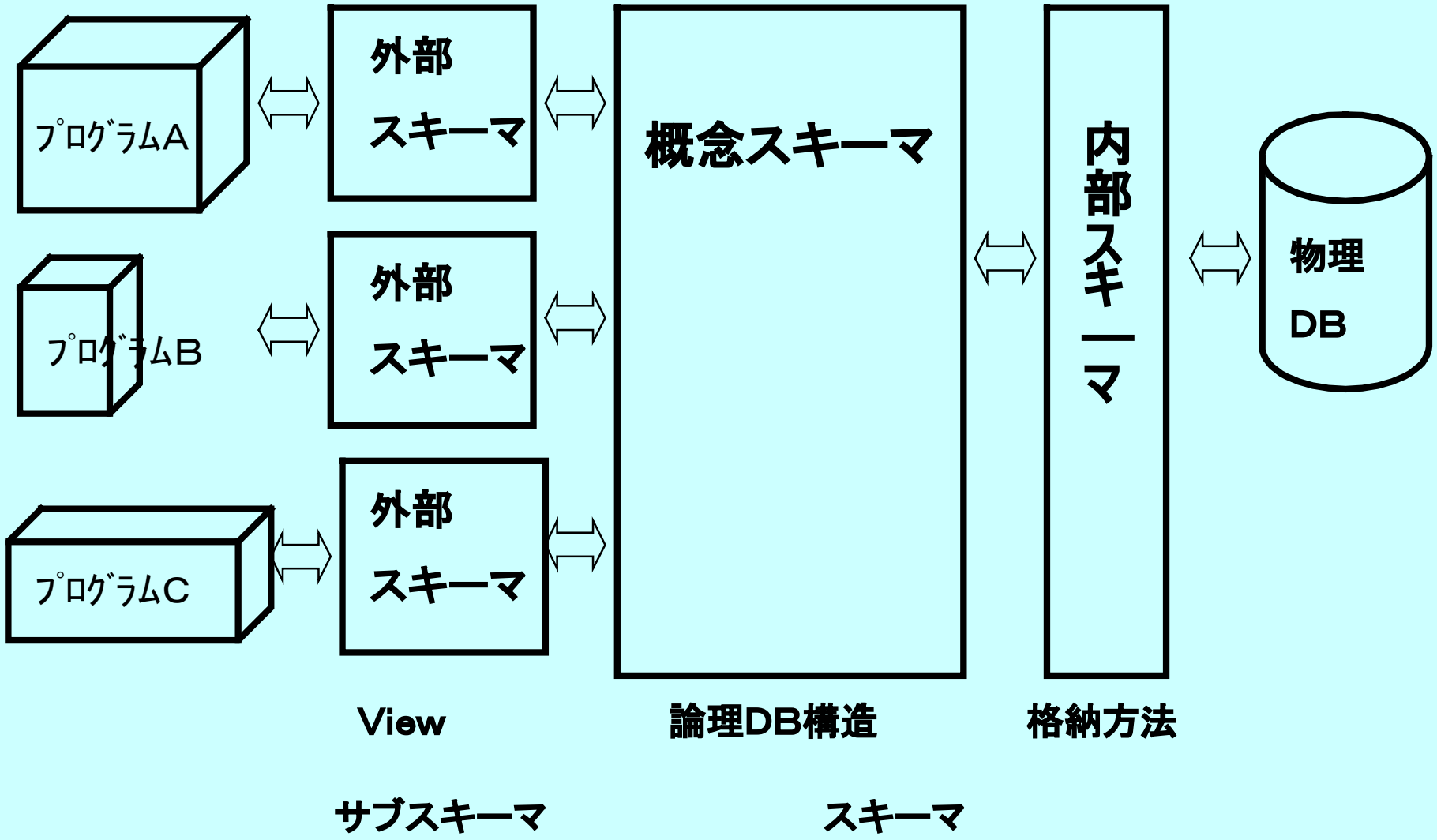
３．２　３層が必要な理由

- DBの物理的な面をプログラムから分離
  －記憶装置、スペース量、物理レコード長、
  ファイル編成方式、・・・・・
- DBの論理的な面をプログラムから分離
  －データ構造、データ間の関連、
  データ名、データ型・・・・・
- ユーザーによるDB操作を容易にする
  －ユーザーに高度なDB専門知識を要求しない
  －データの種々の取出し方を可能としたい

３．３　データベースの3層構造

- 概念スキーマ（conceptual　schema）
  ーデータのモデリング結果を論理構造として
  を定義したもので特定のﾌﾟﾛｸﾞﾗﾑに依存しない。
- 外部スキーマ（external　schema）
  ー個々のﾌﾟﾛｸﾞﾗﾑから見たデータの構造を記述
  するものでViewという。
  ー概念スキーマの変更をプログラムに及ぼさない
  ために設ける。
- 内部スキーマ（internal　schema）
  ー概念スキーマを計算機上に実現する方法
  であり、データベースの物理的な格納方法や
  編成方法を記述するもの。

３．３　（続き、3層構成の図解）

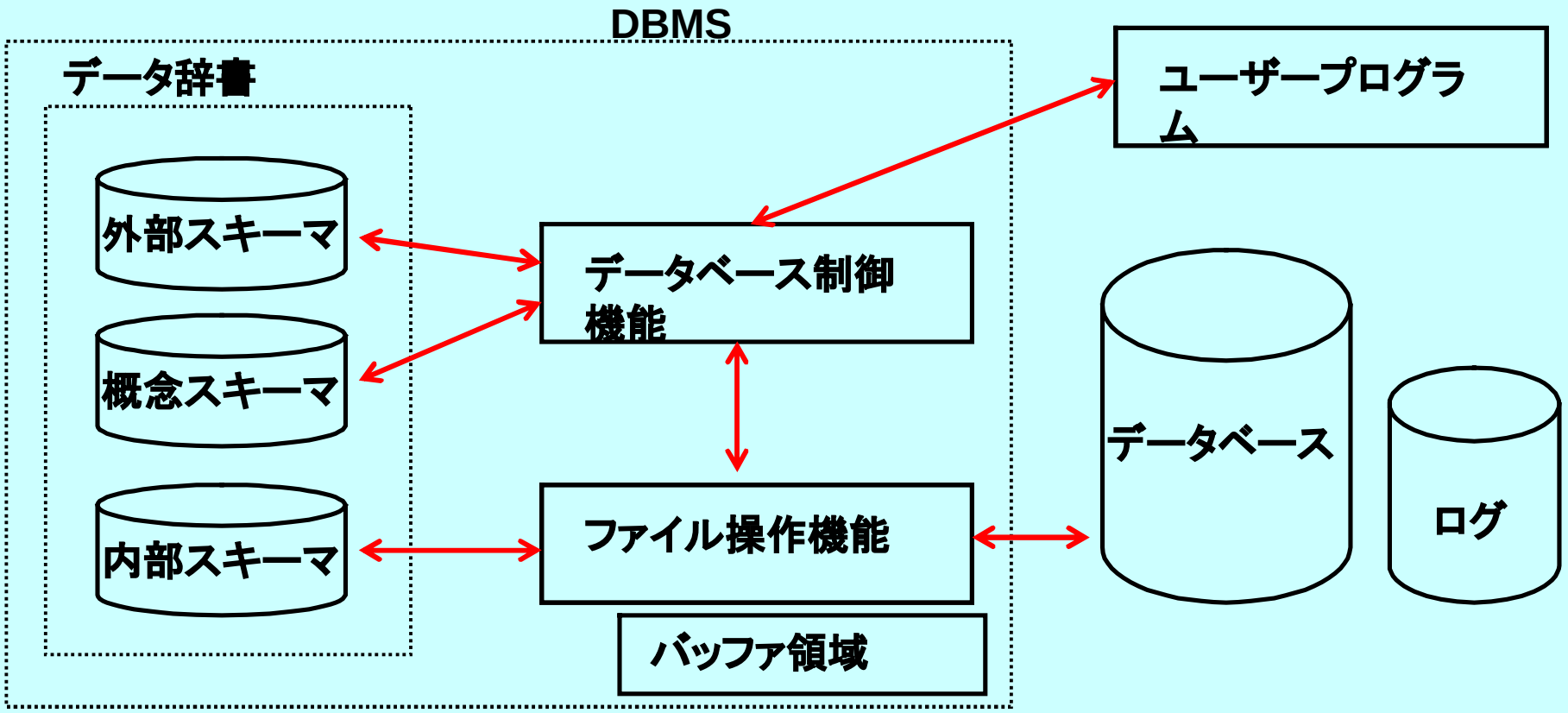


補足:DBMS使用のための言語

- データ記述言語、DDL
  （Data　Description　Language）
  －3つのスキーマを記述する言語
  －データ制御の機能も準備されている
  （同時実行制御、機密保護など）
- データ操作言語、DML
  （Data　Manipulation　Language ）
  －ユーザーがデータアクセスするのために
  使用する言語
  －プログラムに組込む方式（親言語方式）と
  コマンド方式がある（Query　Language）

３．４　DBMSとスキーマの関係

- 各スキーマの記述情報をデータ辞書に集めておき、それをDBMSが使用する。



４．主なファイル編成方式

４．１　予備知識

①ハードディスクの構造

②レコード、ファイル

③ブロック、ギャップ

４．２　順次編成ファイル

４．３　直接編成ファイル

４．４　索引付順次編成ファイル

４．５　木構造索引付ファイル

ファイル中のレコードの並べ方

４．１　予備知識

①　ハードディスクの構造

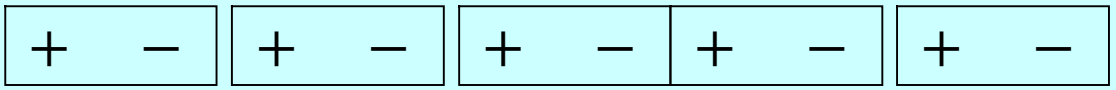
通常、データベースが格納されるハードディスクについて、その概要を述べる。

・呼び名　磁気ディスク、ハードディスク

（DASD-----メインフレームでの用語）

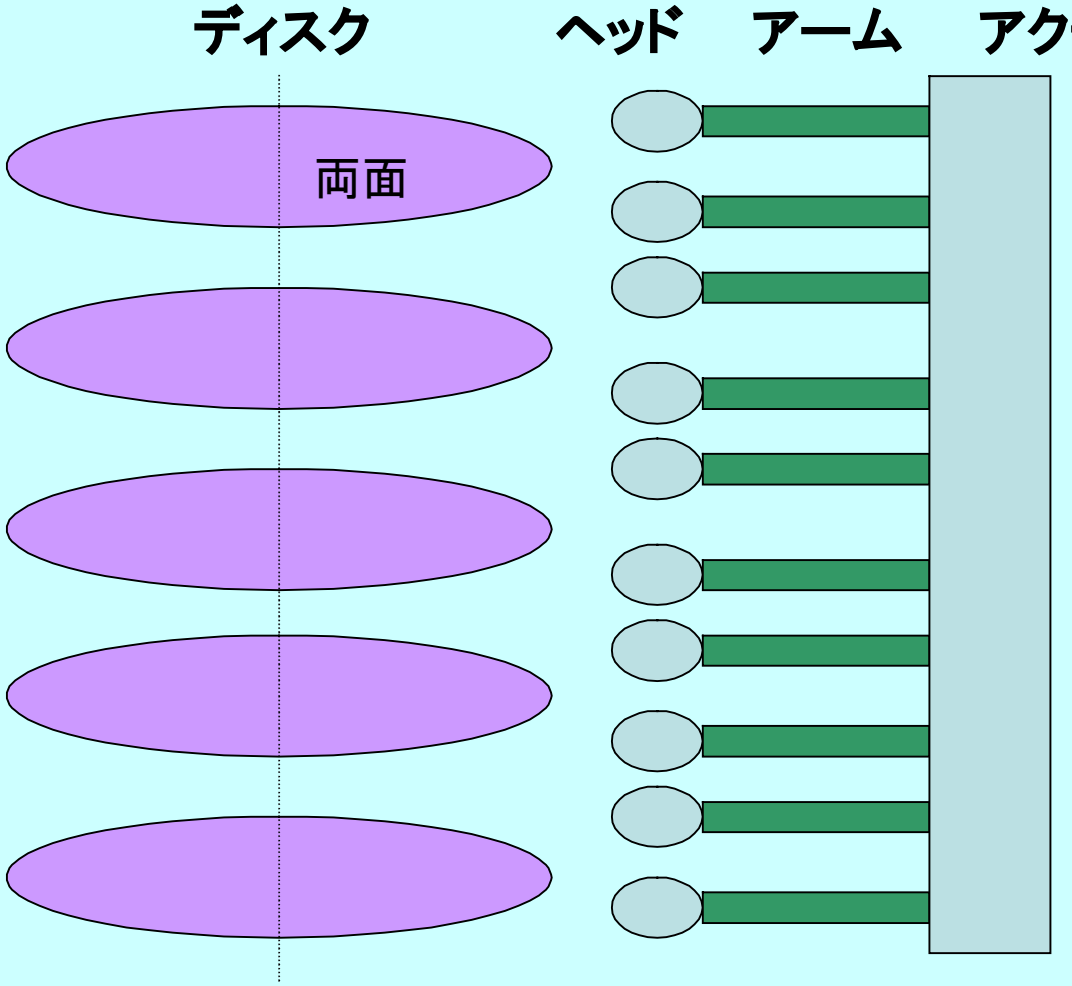
（HDD-------サーバー、パソコンでの用語）

・記録方式　　水平磁気記録



①　（続き）

- 構造



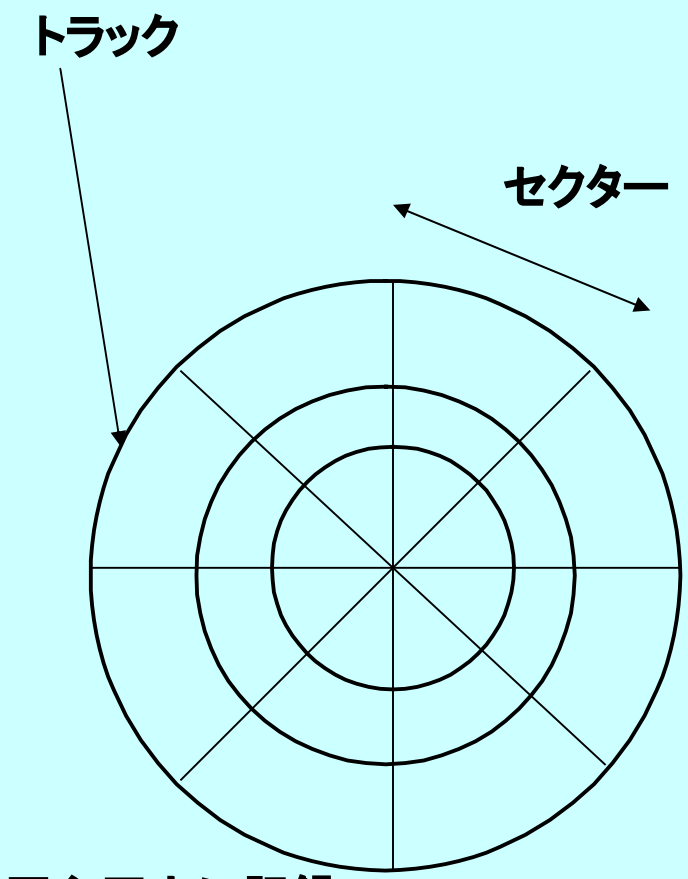
高速回転中のディスクに対して、

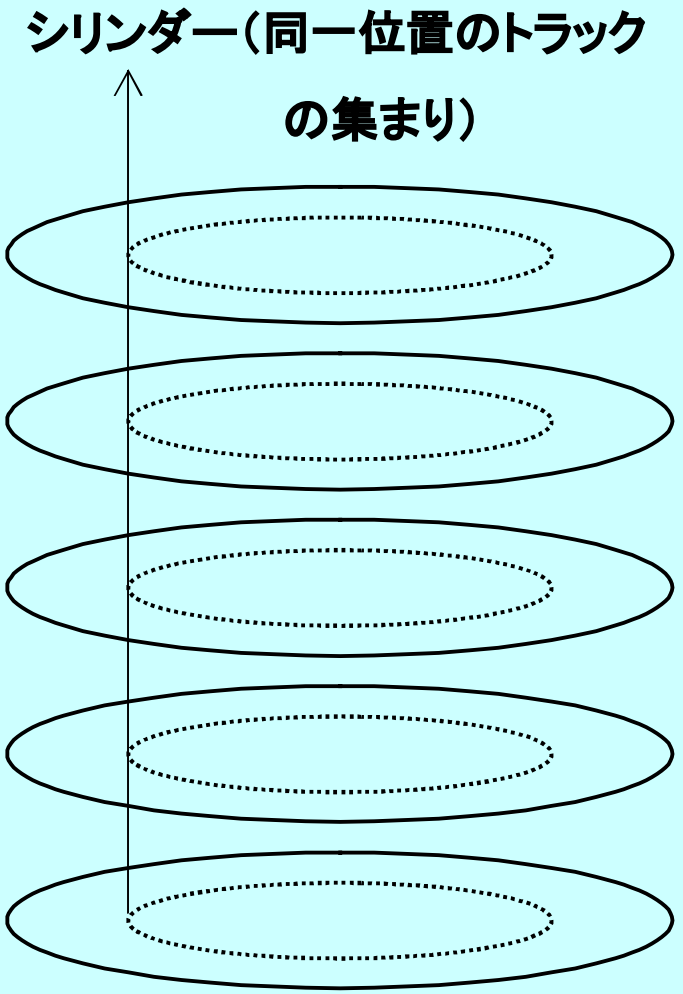
アクチュエータ単位で、全アームが前後に動き、

ヘッドをディスクの所定の位置にとめる。

①　（続き）

- ディスクの使い方





①　（続き）

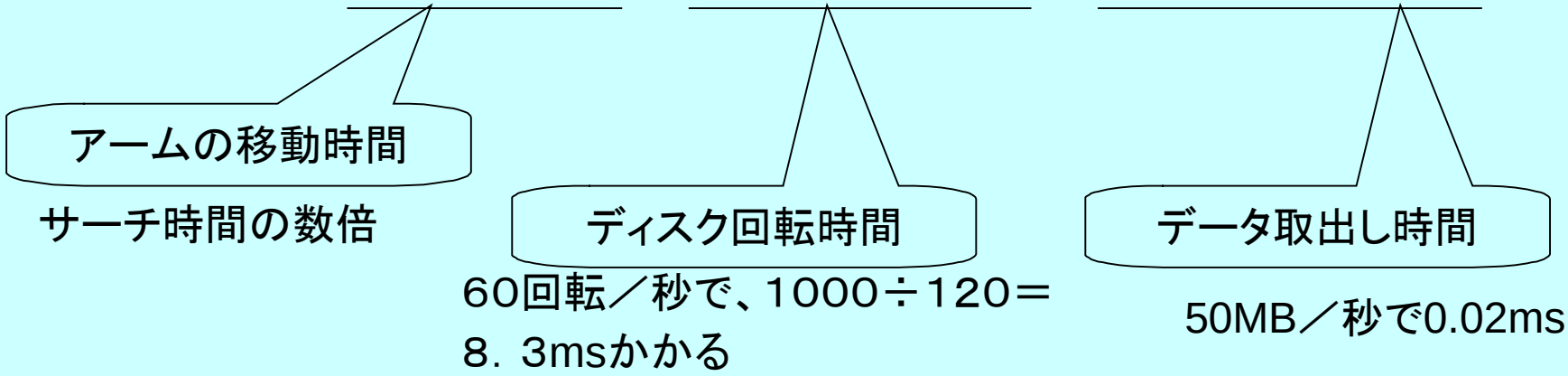
・ディスクのアドレス

アドレス＝ディスク装置番号＋サーフェス番号＋トラック番号

＋セクター番号＋セクター内の相対位置番号

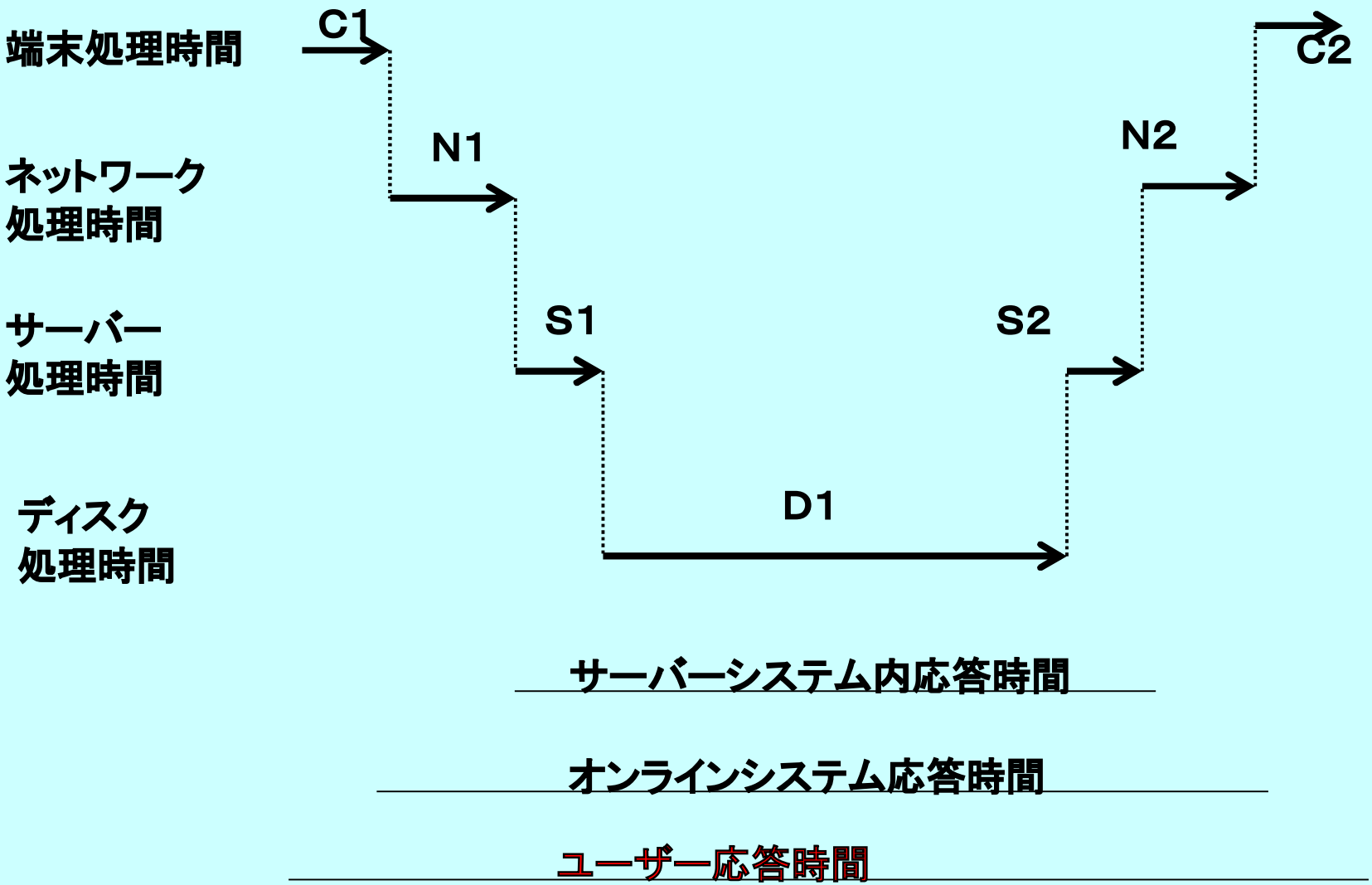
・ディスク検索時間

合計タイム＝シーク時間＋サーチ時間＋データ転送時間



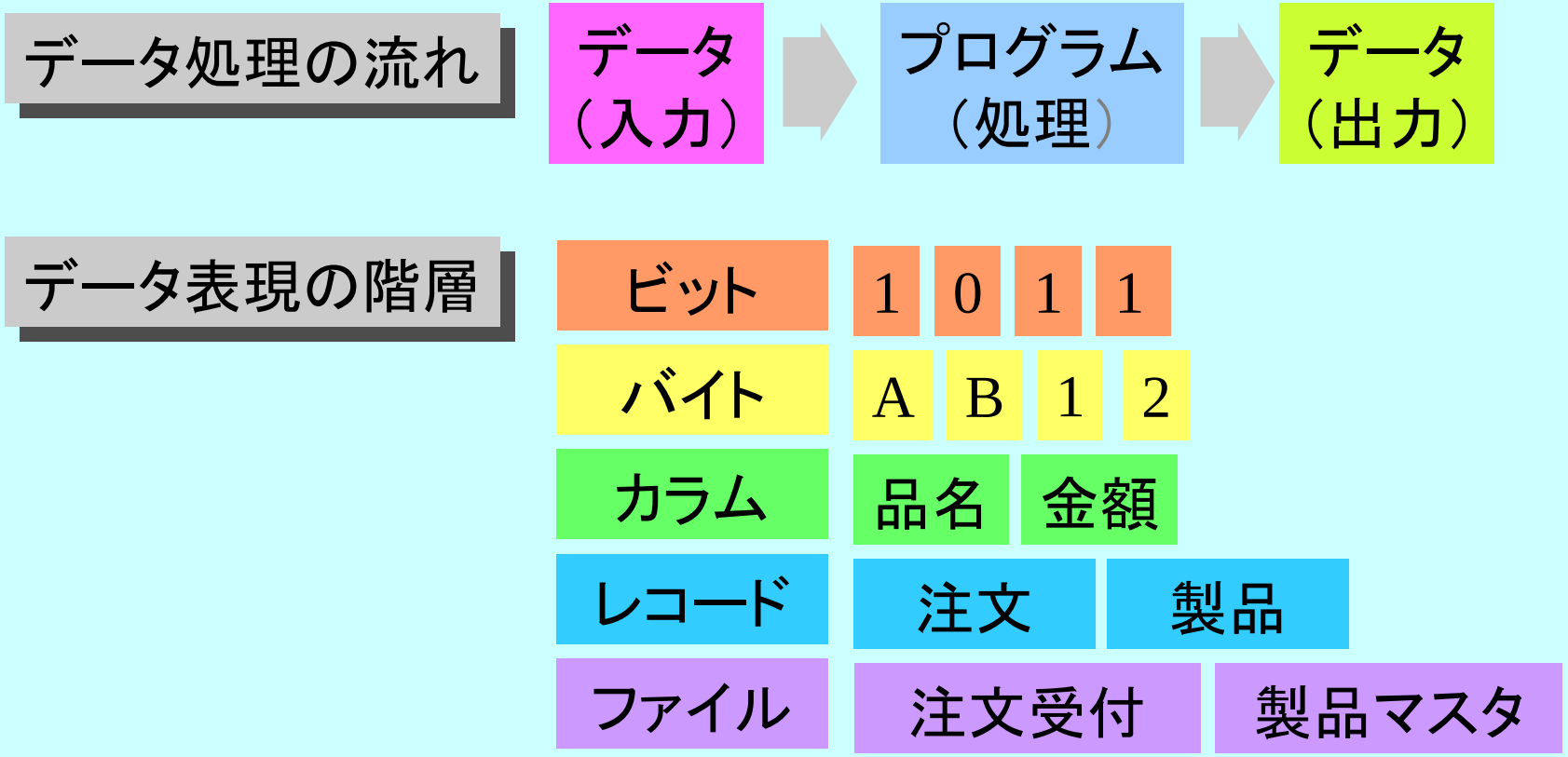
（補足）レスポンスタイムの内訳

例：オンライン処理の場合



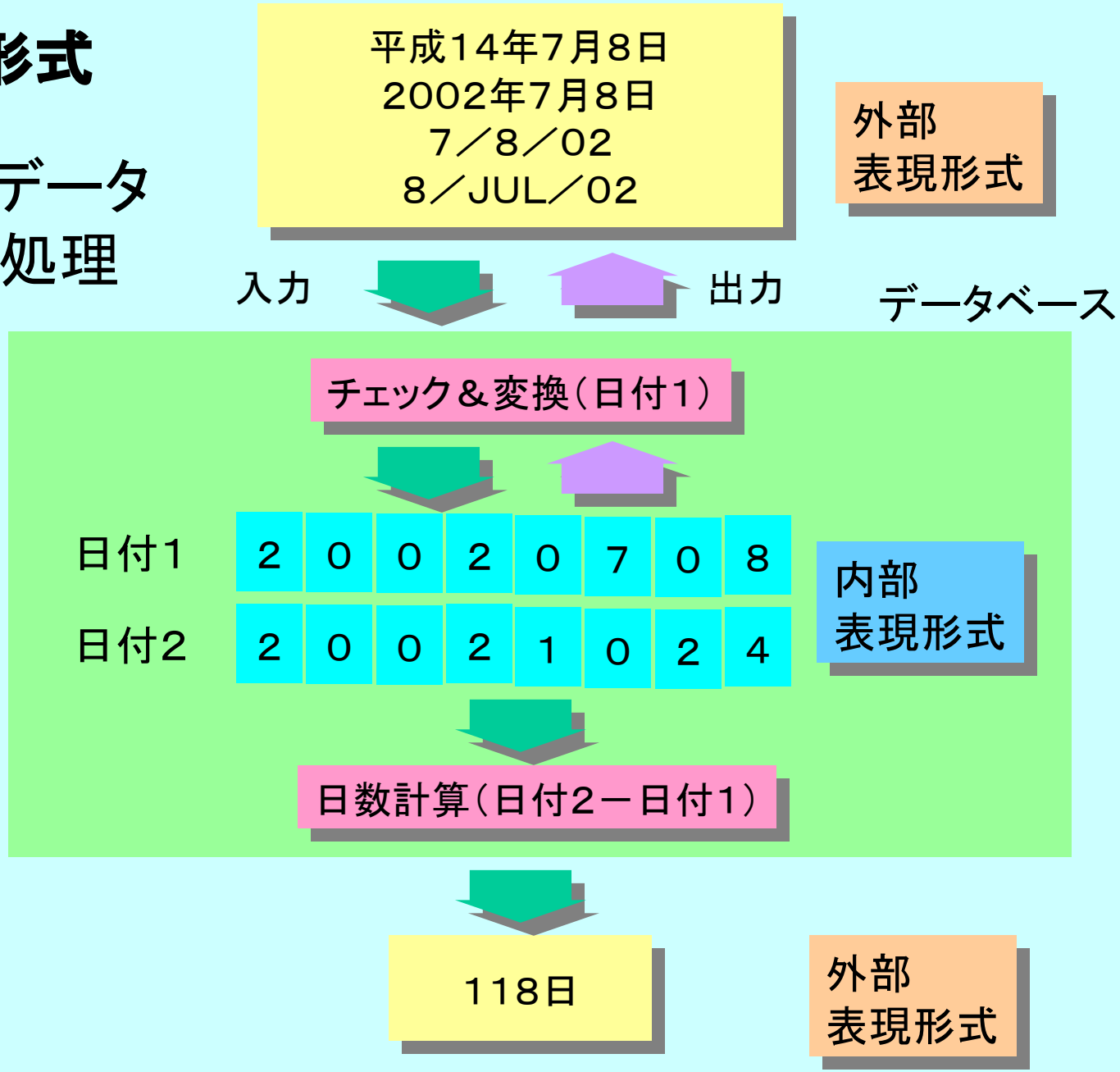
② レコード、ファイル

- コンピュータシステムにおけるデータの扱い



内部表現形式

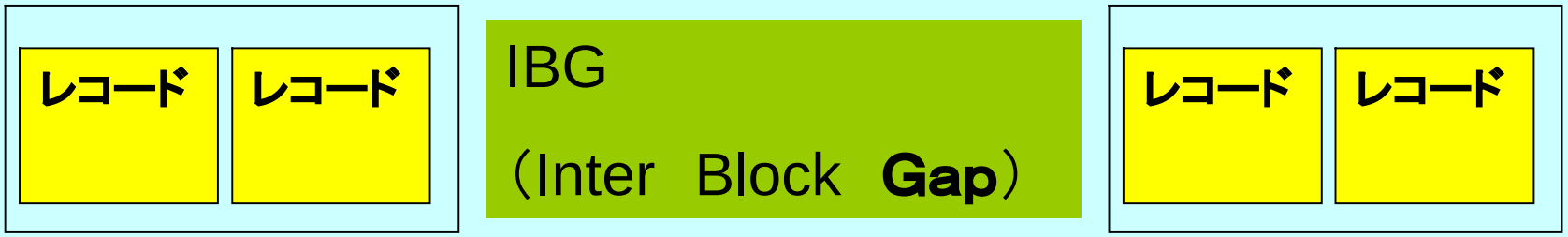
（例）日付データの処理



③　ブロック、ギャップ

- プログラムは、レコード単位で読み書きするが、
- ディスク装置では、読み書きの速度を上げる
  ために、ブロックという単位で読み書きを行う。
  ーブロッキング・ファクター（ブロック化因数）
  ー何件かのレコードを集めてーかたまりに。

・最大は、1シリンダーまたは1セクター



ブロック
（物理レコード）

次のIO命令までの、
ディスク回転距離

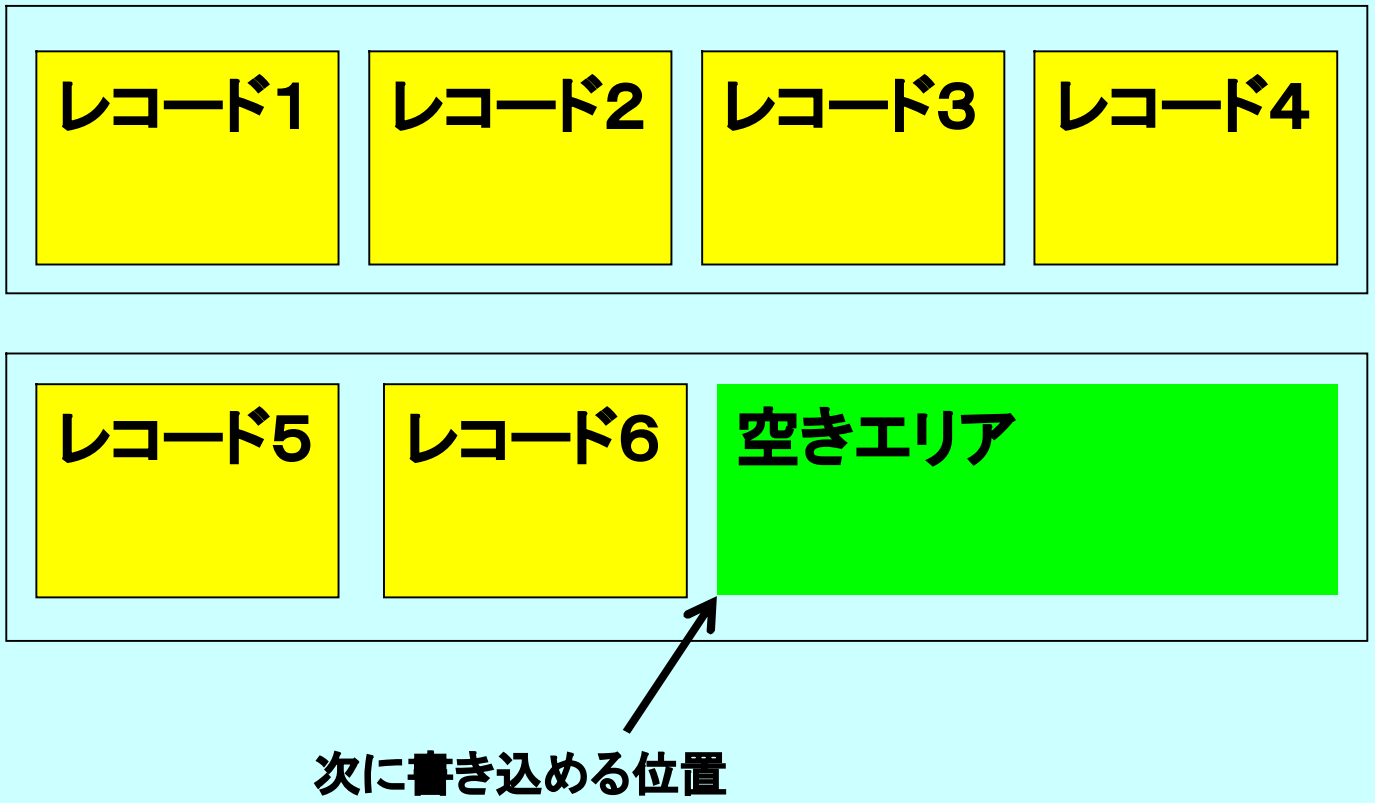
ブロック
（物理レコード）

４．２　順次ファイル編成

- 書き出した順序どおりにデータが記録される。
- 最も古く、基本的なファイル編成。
- レコードはブロック単位に記録される。
- 磁気テープでは、この方式が唯一の方法。
- 長所：操作が簡単、記憶装置の使用効率が高い。
  通常ブロック、バッファーサイズを大きくする。
- 短所：特定のレコードのアクセスに時間がかかる。
  レコードの挿入・削除時は、ファイル全体を書き替え。

Sequential Access Method （SAM）

４．２　　（続き）

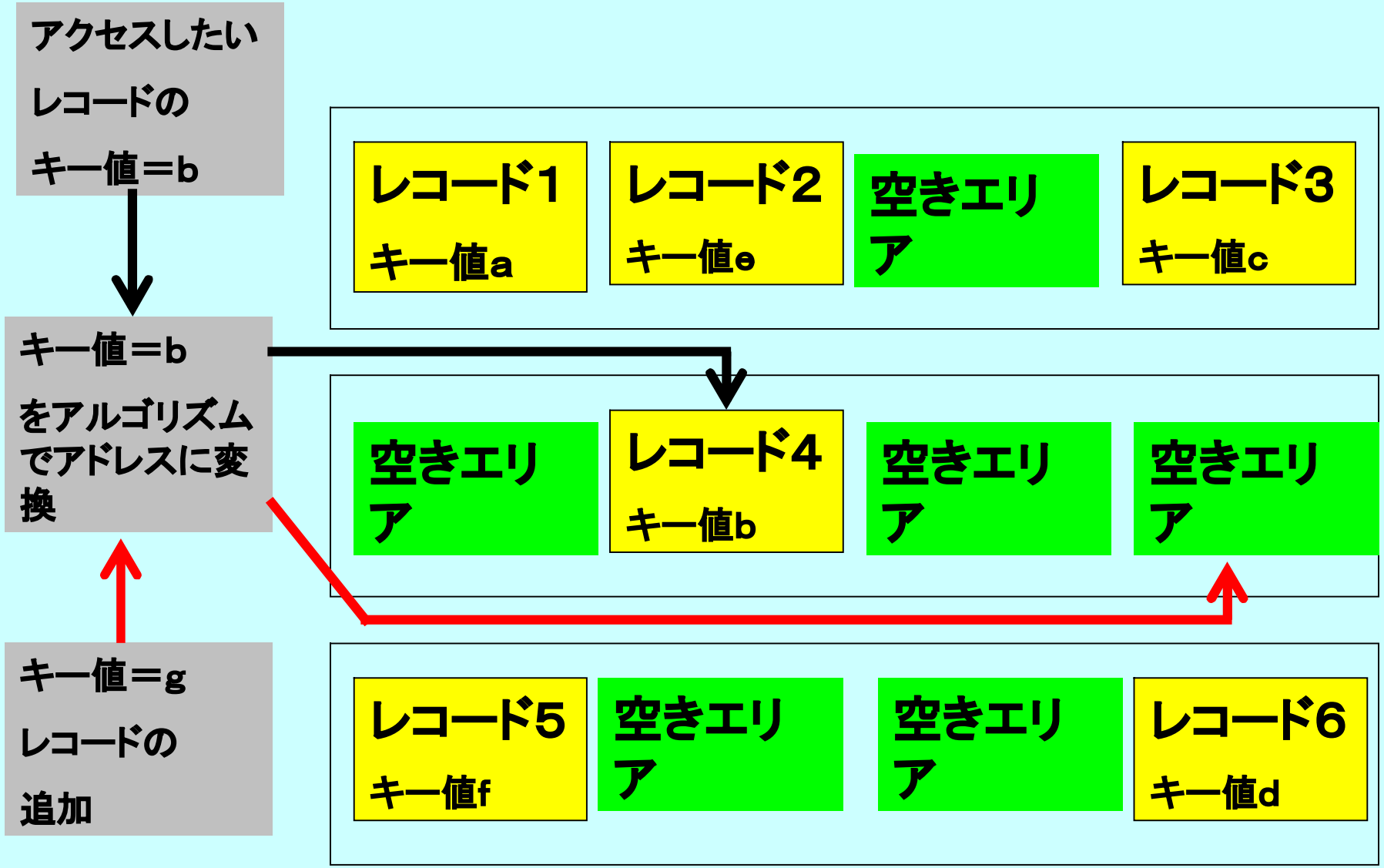


４．３　直接編成ファイル
（ランダム、ハッシュ）

・レコードが持つキーとディスクアドレスを対応付ける。
　キーが分れば、直接的にレコードを取出すことが出来る。
・キー値を、ある一定のアルゴリズムで変換したものを
　ディスクアドレスにする。
・長所：レコードを取出す時間が、極めて短い。
　　通常ブロック、バッファーサイズを大きくする。
・短所：アルゴリズムにより、ダブリが生じ（衝突、シノニム）、
　　その対応が複雑になる。
　　レコードの順次取出しは出来ない。ディスク量が大。

Direct　Access　Method　（DAM）、　　Hash＝コマ切れ

４．３　　（続き）

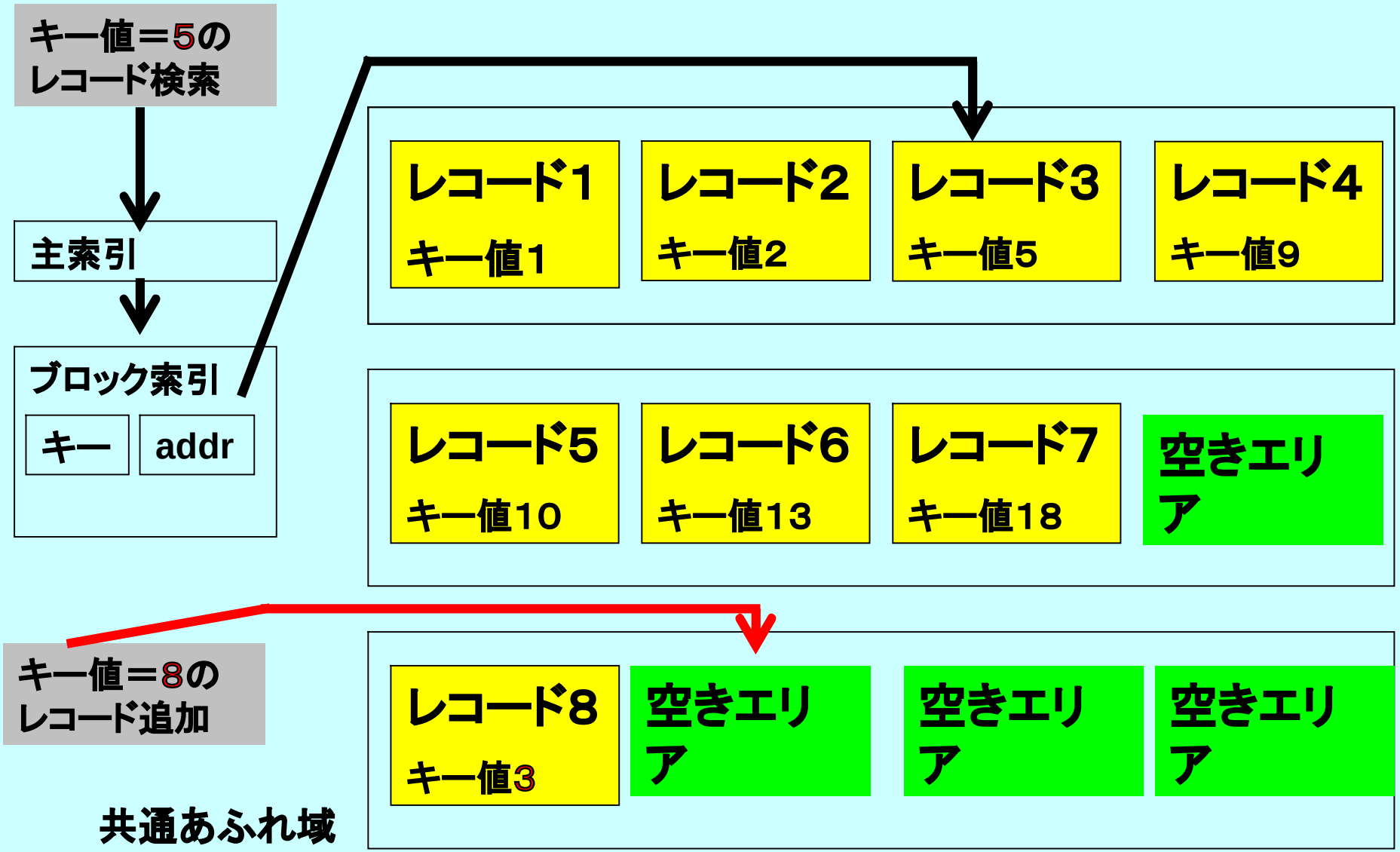

アクセスしたい
レコードの
キー値＝b
キー値＝b
をアルゴリズム
でアドレスに変
換
キー値＝g
レコードの
追加
レコード1
キー値a
レコード2
キー値e
空きエリア
レコード3
キー値c
空きエリア
レコード4
キー値b
空きエリア
空きエリア
レコード5
キー値f
空きエリア
空きエリア
レコード6
キー値d

４．４　索引付順次編成ファイル

- 順次編成ファイルをもとにして、キーとそのレコードが格納されているディスクアドレスを対にした索引を持つ。
- 共有あふれ領域を設けて、順次編成の欠点を補う。
- 長所：順次でもキー値指定でもアクセスできる。
  レコードの挿入/削除にも、それほど時間がかからず
- 短所：共有あふれ領域の確保が必要。
  共有あふれ領域の使用率が高まると、ファイルの再編成が必要となる。

Indexed Sequential Access Method （ISAM）

４．４　　（続き）

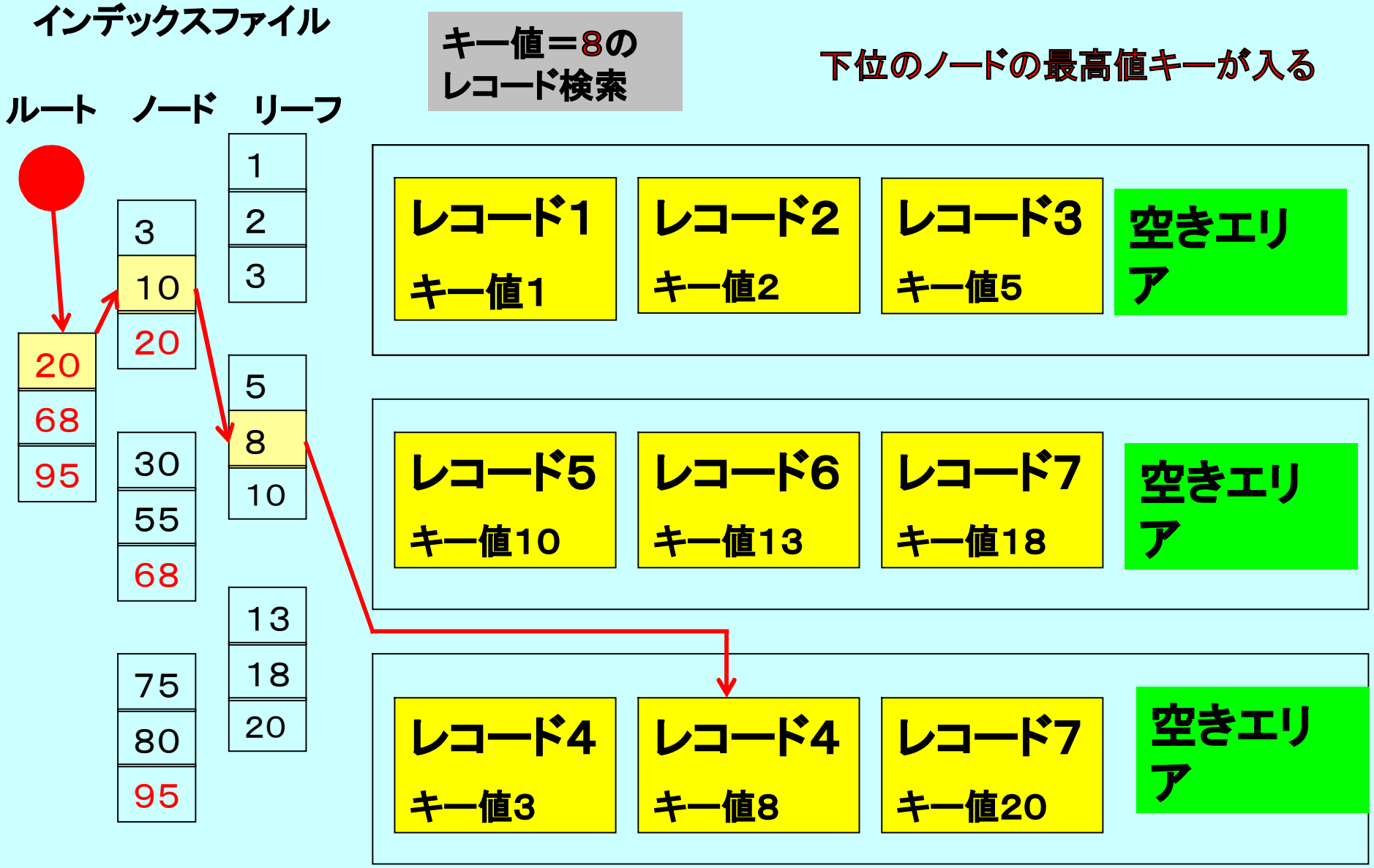

キー値＝5の
レコード検索
主索引
ブロック索引
キー
addr
レコード1
キー値1
レコード2
キー値2
レコード3
キー値5
レコード4
キー値9
レコード5
キー値10
レコード6
キー値13
レコード7
キー値18
空きエリア
キー値＝8の
レコード追加
レコード8
キー値3
空きエリア
空きエリア
空きエリア
共通あふれ域

４．５　木構造索引付ファイル

- 順次編成ファイルをもとに、索引を木構造で作成する
  ことにより、大量データのキー検索を高速化したもの。
- メモリー、ディスクが急速に安価になり、
  かつCPU、ディスクの高速化により可能となった。
- 長所：順次でもキー値指定でもアクセスできる。
  レコードの取出しが高速で出来る。
- 短所：索引のための領域が増大する。
  索引の更新頻度が高まり、オーバーヘッドが大。
  データ量が少ない場合は、効果が薄い。

（リレーショナルデータベースに適したファイル編成）

4．5　（続き）


インデックスファイル
キー値＝8の
レコード検索
下位のノードの最高値キーが入る
ルート
ノード
リーフ
20
68
95
3
10
20
30
55
68
75
80
95
1
2
3
5
8
10
13
18
20
レコード1
キー値1
レコード2
キー値2
レコード3
キー値5
空きエリア
レコード5
キー値10
レコード6
キー値13
レコード7
キー値18
空きエリア
レコード4
キー値3
レコード4
キー値8
レコード7
キー値20
空きエリア

５．レポート課題

①データ独立と3層スキーマについて、説明してください。

②リレーショナルデータベースによく使用されるファイル
編成方式を取り上げて、具体例を記述してください。

①レポートの内容レベルは、A4x1枚程度。

②次回の授業開始時に、提出して下さい 。
（ただし、それ以前に提出する場合は、
メールで願います。
アドレス： fwhy6454@mb.infoweb.ne.jp ）

６．参考書ほか

- 大木幹雄「ﾃﾞｰﾀﾍﾞｰｽ設計の基礎」（日本理工出版会）
- 増永良文「ﾘﾚｰｼｮﾅﾙ・ﾃﾞｰﾀﾍﾞｰｽ入門」（ｻｲｴﾝｽ社）
- 小野哲ほか「**まるごと図解**、SQLがわかる」（技術評論社）
- 情報処理学会「情報処理ハンドブック」（オーム社）
- 織田敬三「**ビジネス・パソコンユーザーのための ネットワーク対応**、データベース構築ガイド」（電波新聞社）
- 仲田　聰ほか「SE**の基礎知識**　コンピュータテクノロジー」（リックテレコム）
- http://www.ann.hi-ho.ne.jp/hirok/sql/index.html
- http://www.rfs.jp/sitebuilder/sql/