# データベース（第7回）

2008.06.09

宇陀則彦

1

## データベースシステム

- データベースシステム＝データベース(DB)＋データベース管理システム（DBMS）

料理と皿（その他、調理器具）

- データベース：複数の応用目的でデータ共有するための、相互に関連づけられた冗長性のないデータ群（統合されたデータ）
- データベース管理システム：データの一貫性チェック、機密保護、同時実行制御、障害回復などを行うソフトウェア

2

## データベース管理システム(DBMS)

- データ記述・操作系
- 整合性維持（各種制約）
- 問合せ処理

} データモデルに依存

- 記憶管理・アクセス機構
- トランザクション管理
  - 同時実行制御
  - 障害回復
- 機密保護
- （エンドユーザインタフェース）

3

## トランザクション管理

- トランザクション：（業務処理、取引の意）
  アプリケーションにおける一連の処理単位

例）

- 銀行の送金処理
- 切符予約処理
- 商品在庫管理処理

4

## 銀行の送金処理

begin
　残高チェック
　update 口座A
　　set 残高=残高－振替金額
　　where 口座番号= 依頼人
　update 口座B
　　set 残高=残高＋振替金額
　　where 口座番号=受取人
end

途中で止まると振替分が消える

5

## 切符予約

begin
　A地点からD地点まで切符予約

　A地点からB地点まで予約
　B地点からC地点まで予約
　C地点からD地点まで予約
end

途中で止まると目的地に行けない

6

## トランザクション管理

- トランザクション：(業務処理、取引の意)
  アプリケーションにおける一連の処理単位
- データベース処理はトランザクションが単位でなければならない。
  (プログラムを書くときなどは常に意識する)
- ACID特性を満たす必要がある。

7

## ACID特性

- Atomicity (原始性)
  トランザクション内の処理は全て有効か全て無効かのどちらかであること
- Consistency (整合性)
  トランザクションの前後ともに正しい状態であること
- Isolation (隔離性)
  他のトランザクションの影響を受けないこと
- Durability (耐久性、持続性)
  障害などで失われないこと

8

## トランザクションの基本命令

- Begin: トランザクションの開始を宣言
- Commit: 処理を確定 (有効に) すること
- Abort: 処理を放棄し、
  元に戻す (ロールバック) すること
- (End): トランザクションの終了を宣言

9

## 同時実行制御

- 通常、複数のトランザクションが同時発生
- 入出力待ちやユーザの入力待ちなどの待ち時間が発生
- 待ち時間中に別の処理が入ると、データベースに異常が発生する
- 異常が起きないように制御する。

10

## データ不整合 (例1)

トランザクション$T_1$
```
begin
    read(A)
    write(A:=A－30)
end
```

トランザクション$T_2$
```
begin
    read(A)
    write(A:=A－20)
end
```

Aの初期値を100とすれば、$T_1$, $T_2$が同時に実行されなければ、どちらの順序で行っても結果は50になる。

11

## データ不整合 (例1)

トランザクション$T_1$
```
begin
    read(A)
                        write(A:=A－30)
end
```

トランザクション$T_2$
```
begin
    read(A)
    write(A:=A－20)
end
```

結果は70となり、$T_2$は反映されない。

12

## データ不整合 (例2)

**トランザクション$T_1$**
begin
　　read(A)
　　read(B)
end

**トランザクション$T_2$**
begin
　　read(A)
　　write(A:=A－20)
　　read(B)
　　write(B:=B+20)
end

A,Bそれぞれ100万あり、AからBに20万円を送金すると、Aは80万円で、Bが120万円になる。

13

## データ不整合 (例2)

**トランザクション$T_1$**
begin

　　read(A)
　　read(B)
end

T1のread(B)は100として読み出される。

**トランザクション$T_2$**
begin
　　read(A)
　　write(A:=A－20)

　　read(B)
　　write(B:=B+20)
end

14

## 直列可能性

- トランザクション$T_1 \ldots T_n$を並列処理したときの結果が、それらを逐次処理した結果と一致すること
- 並行スケジュール (ヒストリ：履歴) が直接可能スケジュールと等価であることを保障する。
  - 競合等価
  - ビュー等価

15

## 隔離性水準

- 他のトランザクションとの影響の度合い
- 3つの異常を定義
  - ダーティリード(P1)
    コミットしていないデータを読むこと
  - 繰り返し不能リード(P2)
    同じデータを複数回読むとき、結果が違うこと
    (タプルAの値が違っている)
  - ファントム(P3)
    異なるデータ集合を読んでしまうこと
    (さっきまであったタプルBがなくなっている)

16

## 隔離性水準

- 他のトランザクションとの影響の度合い

| 隔離性水準 | 現象 (可能性) | | |
|---|---|---|---|
| | P1 | P2 | P3 |
| Read Uncommitted | あり | あり | あり |
| Read Committed | なし | あり | あり |
| Repeatable Read | なし | なし | あり |
| Serializable | なし | なし | なし |

17

## ロック (Lock) による同時実行制御

- 異常が発生しないように、あるトランザクションが実行の間、対象項目をロックし (鍵をかけ)、他のトランザクションは待たせる。

| | 共有ロック | 専有ロック |
|---|---|---|
| 共有ロック | ○ | × |
| 専有ロック | × | × |

18

## デッドロック

- お互いがロックをかけあい、待ちの状態になること

| | |
|---|---|
| $T_1$ | lock A |
| $T_2$ | lock B |
| $T_1$ | lock B |
| | （$T_1$が待ち状態） |
| $T_2$ | lock A |
| | （$T_2$が待ち状態） |

19

## 障害回復

- 障害の種類
  - トランザクション障害
    トランザクションが異常終了する場合
  - システム障害
    システムダウン（ディスク障害除く）
  - メディア障害
    ディスクに障害
- 基本的にはログ（記録）を用いる。
  - 全てのトランザクションの記録をとっておく。

20

## データベース管理システム(DBMS)

- データ記述・操作系
- 整合性維持（各種制約）
- 問合せ処理

} データモデルに依存

- 記憶管理・アクセス機構
- トランザクション管理
  - 同時実行制御
  - 障害回復
- 機密保護
- （エンドユーザインタフェース）

21

## アクセス法（物理構造）

- ファイルアクセス法
  - ディスクのどこに目的のデータがあるか
    - リレーション → ファイル
    - タップル → レコード
    - 属性 → フィールド
- ファイル編成法
  - ファイルのデータをどのように書いておくか

22

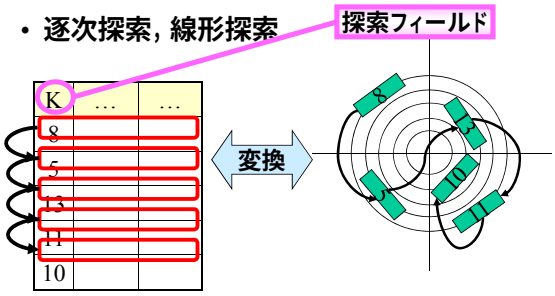
## ファイルアクセス法

- スキャン (scan)
- 探索 (search)
- インデックス法 (indexing)
- ハッシュ法 (hashing)

23

## スキャン (scan)

- 逐次探索，線形探索




24

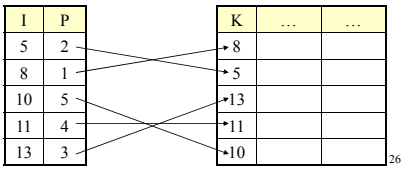
## 探索 (search)

- 探索キーが整列されている (仮定)
  - 二分探索
    - 半分の半分？
    - 二分して目的のデータを見つける
  - ブロック探索
    - ある間隔でデータを区切り，目的のデータを見つける

25

## インデックス法 (indexing)

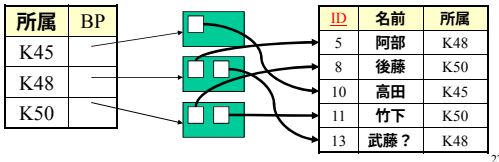
- レコードを同定するポピュラーな方法
- 基本的には
  - 索引レコードとポインタレコードからなるファイル
  - 索引レコードは整列済み
  - ポインタレコードはデータレコードへのポインタ




26

## インデックス法 (indexing)

- 1次インデックス
  - インデックスとデータレコードが1対1対応
- 2次インデックス
  - インデックスとデータレコードが1対多対応
  - ポインタレコードの値はブロックポインタ




27

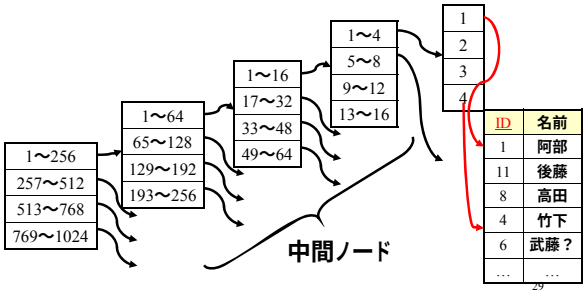
## 多段インデックス

- インデックスのインデックス？
  - ISAM
  - B-tree, B+tree
  - R-tree
  - VSAM

28

## 多段インデックス

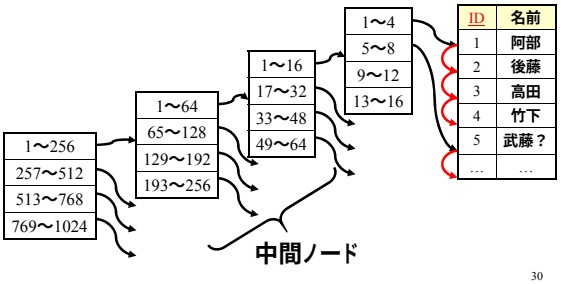
- Tree 構造をもつ索引 (非順次ファイル)




29

## ISAM
## (Indexed Sequential Access Method)

- Tree 構造をもつ索引 (順次ファイル)




30

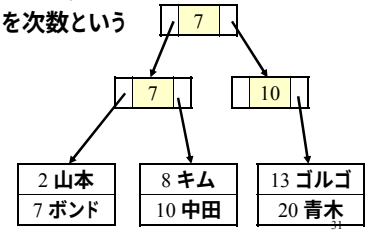
# ISAM

| $p_1$ | $k_1$ | $p_2$ | $k_2$ | … | $p_{q-1}$ | $k_{q-1}$ | $p_q$ |
|---|---|---|---|---|---|---|---|

$k_1 < k_2 < \cdots < k_{q-1}$

$q \le d$ で，$d$ を次数という

| ID | 名前 |
|---|---|
| 2 | 山本 |
| 7 | ボンド |
| 8 | キム |
| 10 | 中田 |
| 13 | ゴルゴ |
| 20 | 青木 |

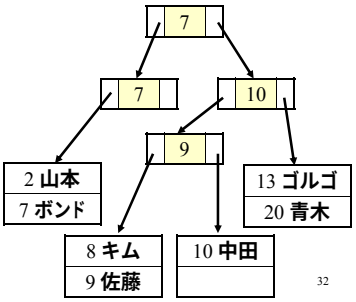



31

# ISAM

- (9, 佐藤) が挿入されたら

| ID | 名前 |
|---|---|
| 2 | 山本 |
| 7 | ボンド |
| 8 | キム |
| 9 | 佐藤 |
| 10 | 中田 |
| 13 | ゴルゴ |
| 20 | 青木 |




32

# B-tree (d次のB-tree)

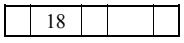
| $P_1$ | $R_1$ | $P_2$ | $R_2$ | … | $P_i$ | $R_{i+1}$ | 以下空き領域 |
|---|---|---|---|---|---|---|---|

- $i$ について
  - $i$ がrootノード以外で $d \le i \le 2d$
  - $i$ がrootノードで $2 \le i \le 2d$
- レコード$R$の〈探索〉キー値を$Key\langle R\rangle$とすると
  $Key(R_1) < Key(R_2) < \cdots < Key(R_i)$
  - レコードは最大$2d$個まで
- $P_j$中の$R$について
  - $j = 1$ならば，$Key(R) < Key(R_i)$
  - $1 < j < i+1$ならば，$Key(R_{j-1}) < Key(R) < Key(R_j)$
  - $j = i+1$ならば，$Key(R_i) < Key(R)$

33

# B-tree

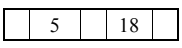
- 18，5, 25, 33，28, 10，22, 13, 16, 7 の順にデータが入力されるとき




34

# B-tree

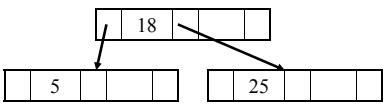
- 18，5, 25, 33，28, 10，22, 13, 16, 7 の順にデータが入力されるとき




35

# B-tree

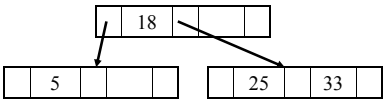
- 18，5, 25, 33，28, 10，22, 13, 16, 7 の順にデータが入力されるとき




36

## B-tree

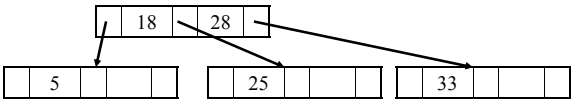
- **18, 5, 25, 33, 28, 10, 22, 13, 16, 7 の順にデータが入力されるとき**




37

## B-tree

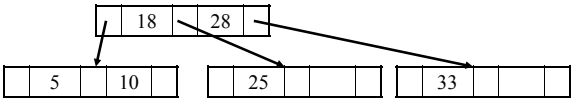
- **18, 5, 25, 33, 28, 10, 22, 13, 16, 7 の順にデータが入力されるとき**




38

## B-tree

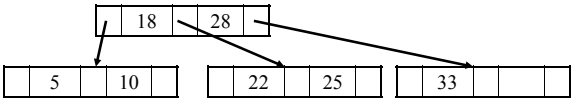
- **18, 5, 25, 33, 28, 10, 22, 13, 16, 7 の順にデータが入力されるとき**




39

## B-tree

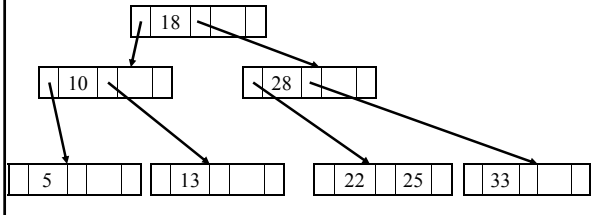
- **18, 5, 25, 33, 28, 10, 22, 13, 16, 7 の順にデータが入力されるとき**




40

## B-tree

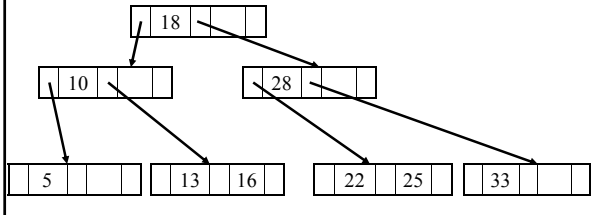
- **18, 5, 25, 33, 28, 10, 22, 13, 16, 7 の順にデータが入力されるとき**




41

## B-tree

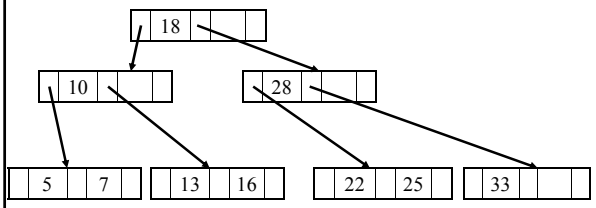
- **18, 5, 25, 33, 28, 10, 22, 13, 16, 7 の順にデータが入力されるとき**




42

## B-tree

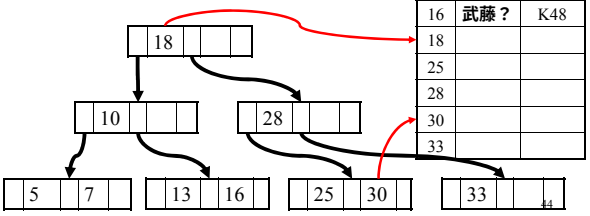
- **18, 5, 25, 33, 28, 10, 22, 13, 16, 7 の順にデータが入力されるとき**




43

## B-tree

- **子ノードは最大で**2d+1




| ID | 名前 | 所属 |
|---|---|---|
| 5 | 阿部 | K48 |
| 7 | 後藤 | K50 |
| 10 | 高田 | K45 |
| 13 | 竹下 | K50 |
| 16 | 武藤？ | K48 |
| 18 | | |
| 25 | | |
| 28 | | |
| 30 | | |
| 33 | | |

44