

データベースシステム(12)(13)

問い合わせの高速化・最適化
障害回復
セキュリティ
その他

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

1

データベース管理システム (Database Management System)

———「DBMS」と呼ぶ

一般的に、(巨大な)統合化されたデータ集合を扱う

データは

実世界(のビジネス)をモデル化したもの

実体 (「学生」, 「科目」など)

関係 (「上戸彩」は「情報構造論」を履修)

DBMSはこうしたデータを保管し、管理するために
デザインされたソフトウェアパッケージ

なぜデータベースを学ぶのか？

データの多様性、サイズともに増加
そして情報を制するものが勝つ時代

デジタルライブラリ、
インタラクティブビデオ、
ヒトゲノム計画など

DBMS
による管理が必要



DBMSは巨大なデータを管理したり、
データを効率的に参照、変更、追加、削除するために
利用される

障害回復機能、同時実行制御、アクセス制御などで
有益な機能をもつ。(他にもデータ独立性、
データ一貫性の保障など重要な機能をもっている)

DBMSの仕組みを学ぶことで、情報の構造
情報管理、情報検索の真髄が理解できる

データを表形式で表現するデータモデル「関係モデル」
関係モデルのデータを管理するためのソフトウェアが
「関係データベース」(問い合わせなどの機能をサポート)

データベース管理システム (DBMS, Database Management System)

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

4

ファイル vs DBMS

(多数のユーザが利用する)データの
不一致や矛盾を防ぐ

「一貫性制約」
「同時実行制御」
「トランザクション管理」

クラッシュからの復元

大量データに対する高速アクセス、高速移動
(インデックス、問い合わせ最適化、バッファリング、
ミラーリング、マテリアライズドビュー)

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

5

大規模な表から必要なデータを取り出す

ファイルを対象にプログラムを組む場合、
アルゴリズムとその計算量に細心の注意を払う必要がある

DBMSにデータがある場合、必要なデータをSQLで指定すれば、
DBMSが最適で効率的な方法でデータを取り出してくれる

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

6

SQL問い合わせの(概念的)評価手順

「103番のボートを予約した船乗りの名前を挙げよ」

```
SELECT S.sname
FROM Sailors S, Reserves R
WHERE S.sid=R.sid AND R.bid=103
```

Sailors	sid	sname	rating	age
	22	dustin	7	45.0
	31	lubber	8	55.5
	58	rusty	10	35.0

Reserves	sid	bid	day
	22	101	10/10/96
	58	103	11/12/96

(sid)	sname	rating	age	(sid)	bid	day
22	dustin	7	45.0	22	101	10/10/96
22	dustin	7	45.0	58	103	11/12/96
31	lubber	8	55.5	22	101	10/10/96
31	lubber	8	55.5	58	103	11/12/96
58	rusty	10	35.0	22	101	10/10/96
58	rusty	10	35.0	58	103	11/12/96

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

77

SQL問い合わせの(概念的)評価手順

```
SELECT [DISTINCT] <SELECTリスト>
FROM <FROMリスト>
WHERE <選択条件>
```

- (1) <FROMリスト>のテーブルの直積(cross-product)を計算
- (2) <選択条件>を各行に適用し, 偽ならばその行を削除
- (3) <SELECTリスト>にない属性を削除
- (4) もしDISTINCTが指定されているなら重複行を除く

この順番で得られる答えがSQL問い合わせの答えとなるが, 一般的にこの手順は効率的ではない!
DBMSの最適化機能は同じ答えを求める
「より効率的な手順で」SQL問い合わせを処理する

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

8

計算効率について考えた場合,
前述の「問い合わせの処理手順」は,
同じ結果を得られる, いくつかの手順のなかでほぼ最悪

SQL問い合わせの処理

- (1) <FROMリスト>のリレーションに対し直積を計算
入力が n 行のテーブル m 個なら $O(n^m)$ の計算コスト
出力データのサイズも $O(n^m)$
- (2) <選択条件>を各タプルに適用し, 偽ならば削除
入力が n 行のテーブルなら $O(n)$, 出力データは $O(m)$
 $n \gg m$ のことが多い
- (3) <SELECTリスト>にない属性を削除
入力が n 行のテーブルなら $O(n)$, 出力データは $O(n)$
1行あたりのデータ量が減るので係数や定数が小さくなる
- (4) もしDISTINCTが指定されているなら重複行を除く
入力が n 行のテーブルなら $O(n \log n)$, 出力データは $O(m)$
 $n \gg m$

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

9

船乗り (Sailors)

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

予約 (Reserves)

sid	bid	day
22	101	10/10/96
58	103	11/12/96

「103番のボートを予約した船乗りの名前を挙げよ」

```
SELECT S.sname
FROM Sailors S, Reserves R
WHERE S.sid=R.sid AND R.bid=103
```

(sid)	sname	rating	age	(sid)	bid	day
22	dustin	7	45.0	22	101	10/10/96
22	dustin	7	45.0	58	103	11/12/96
31	lubber	8	55.5	22	101	10/10/96
31	lubber	8	55.5	58	103	11/12/96
58	rusty	10	35.0	22	101	10/10/96
58	rusty	10	35.0	58	103	11/12/96

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

100

不要なコストの削減方法(1)

where条件の中で単独のテーブル内で真偽が判定
できる項は最初に評価する

船乗り (Sailors)

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

予約 (Reserves)

sid	bid	day
22	101	10/10/96
58	103	11/12/96

```
SELECT S.sname
FROM Sailors S, Reserves R
WHERE S.sid=R.sid AND R.bid=103
```

sid	bid	day
58	103	11/12/96

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

111

不要なコストの削減方法(2)

whereにもselectにも現れない属性は最初から不要

船乗り (Sailors)

sid	sname	
22	dustin	
31	lubber	
58	rusty	

予約 (Reserves)

sid	bid	
22	101	
58	103	

```
SELECT S.sname
FROM Sailors S, Reserves R
WHERE S.sid=R.sid AND R.bid=103
```

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

122

```
SELECT S.sname
FROM Sailors S, Reserves R
WHERE S.sid=R.sid AND R.bid=103
```

船乗り (Sailors)

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

sid	sname
22	dustin
31	lubber
58	rusty

予約 (Reserves)

sid	bid	day
22	101	10/10/96
58	103	11/12/96

sid	bid	day
58	103	11/12/96

sid
58

(sid)	sname	(sid)
22	dustin	58
31	lubber	58
58	rusty	58

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

133

```
SELECT S.sname
FROM Sailors S, Reserves R
WHERE S.sid=R.sid AND R.bid=103
```

(1) <FROMリスト>のリレーションに対し直積を計算

入力がn行のテーブルm個なら $O(n^m)$ の計算コスト

出力データのサイズも $O(n^m)$

直積の計算はとても高コスト

(n行のテーブル2つの直積でも $O(n^2)$)

1,000行だと1,000,000

通常は, この例のようにイクイジョイン

最終結果は多くともn行

(改善の余地は多く)

効率的な計算方法が考案されている)

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

144

例題

「さくら組」と「タンボポ」の
両方に所属したモーニング娘の
メンバーを求めよ

```
Select S.名前
From さくら組 S, タンボポ T
Where S.名前 = T.名前
```

さくら組

名前	血液型	誕生日	出身
あべ	A	81/8/10	北海道
やぐち	A	83/1/20	神奈川
よしざわ	O	85/4/12	埼玉
かご	AB	88/2/7	奈良
たかはし	A	86/9/14	福井
こんの	B	87/5/7	北海道
にいがき	B	88/10/20	神奈川
かめい	AB	88/12/23	東京

タンボポ

名前	血液型	誕生日	出身
いしかわ	A	85/1/19	神奈川
こんの	B	87/5/7	北海道
にいがき	B	88/10/20	神奈川
しばた	O	84/2/22	神奈川

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

155

愚者のイクイジョイン

```
Select S.名前
From さくら組 S, タンボポ T
Where S.名前 = T.名前
```

Sの各行に対して
Tの同じ名前の行を調べる

$O(n^2)$

厳密には

S, Tがそれぞれn行, m行
だとしたらn × m回の比較

さくら組 S

名前	...
あべ	...
やぐち	...
よしざわ	...
かご	...
たかはし	...
こんの	...
にいがき	...
かめい	...

タンボポ T

名前	...
いしかわ	...
こんの	...
にいがき	...
しばた	...

こんの	...
にいがき	...

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

166

賢者のイクイジョイン (ソートマージジョイン Sort-Merge Join)

さくら組 S

名前	...
あべ	...
やぐち	...
よしざわ	...
かご	...
たかはし	...
こんの	...
にいがき	...
かめい	...

タンボポ T

名前	...
いしかわ	...
こんの	...
にいがき	...
しばた	...

(1) SとTをそれぞれソート

$O(n \log n)$

(2) S, Tそれぞれにポインタを
を用意し, SとTのそれぞれの
先頭を指す

(3) ポインタの状態から
以下のいずれかを実行
・S=Tならば結果表に追加し
両方のポインタを1行進める

・S < TならばSを1行進める
・S > TならばTを1行進める

(4) S, T両方とも終わりでない
ならば(3)へ戻る

あべ	...	→	いしかわ	...
かご	...	→	こんの	...
かめい	...	→	しばた	...
こんの	...	→	にいがき	...
たかはし	...	→		
にいがき	...	→		
やぐち	...	→		
よしざわ	...	→		

こんの	...
にいがき	...

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

177

効率的な探索アルゴリズムを
使った高速化

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

18

DBMS内部のデータ

```
Create Table 成績表 (
  名前 char(20),
  国語 integer,
  算数 integer,
  英語 integer,
  primary key(名前)
)
```

```
Insert Into
  成績表(名前,国語,算数,英語)
Values(かこ,85,32,80)
```

```
Insert Into
  成績表(名前,国語,算数,英語)
Values(いしかわ,32,70,75)
```

Insertした順に行がテーブルに追加される

成績表

名前	国語	算数	英語
かこ	85	32	80
いしかわ	32	70	75
こんの	30	30	100
やぐち	28	99	82
おがわ	90	10	42
よしざわ	76	45	80
かめい	75	92	40
たかはし	85	20	50
あべ	45	56	98
つじ	20	60	75
にいがき	82	42	75
ふじもと	18	70	78
みちしげ	60	46	88
いいだ	70	75	42
たなか	80	38	68

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

19

問い合わせ文の処理

```
Select *
From 成績表
Where 名前 = 'こんの'
```

テーブルの最初の行から
順番に条件をチェックして探す

この例の場合,
「名前」は主キーなので
条件に一致する行は
多くとも1つしかない。
(行が見つかったら処理を終了)

この処理の計算量は
行数をnとすると $O(n)$

成績表

名前	国語	算数	英語
かこ	85	32	80
いしかわ	32	70	75
こんの	30	30	100
やぐち	28	99	82
おがわ	90	10	42
よしざわ	76	45	80
かめい	75	92	40
たかはし	85	20	50
あべ	45	56	98
つじ	20	60	75
にいがき	82	42	75
ふじもと	18	70	78
みちしげ	60	46	88
いいだ	70	75	42
たなか	80	38	68

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

20

問い合わせ文の処理の効率化

```
Select *
From 成績表
Where 名前 = 'こんの'
```

問い合わせの条件に使われる属性に
(この例の場合, 「名前」に)
インデックスを作成すると
問い合わせの処理が効率的になる

```
Create Index <インデックス名>
```

```
On <テーブル名>
```

```
(<属性名> [, ...])
```

```
Create Index 名前インデックス
```

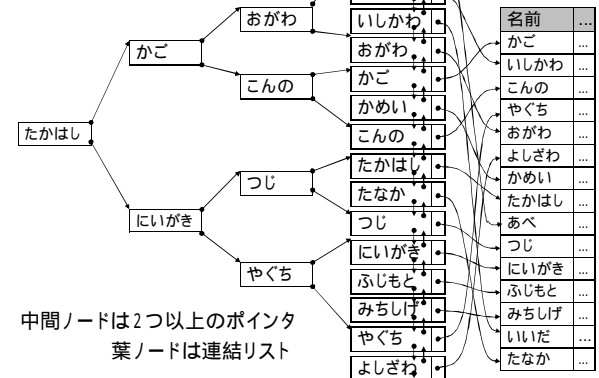
```
On 成績表 (名前)
```

成績表

名前	国語	算数	英語
かこ	85	32	80
いしかわ	32	70	75
こんの	30	30	100
やぐち	28	99	82
おがわ	90	10	42
よしざわ	76	45	80
かめい	75	92	40
たかはし	85	20	50
あべ	45	56	98
つじ	20	60	75
にいがき	82	42	75
ふじもと	18	70	78
みちしげ	60	46	88
いいだ	70	75	42
たなか	80	38	68

B+ツリー (Balanced plus tree)

代表的なインデックスの一つ。

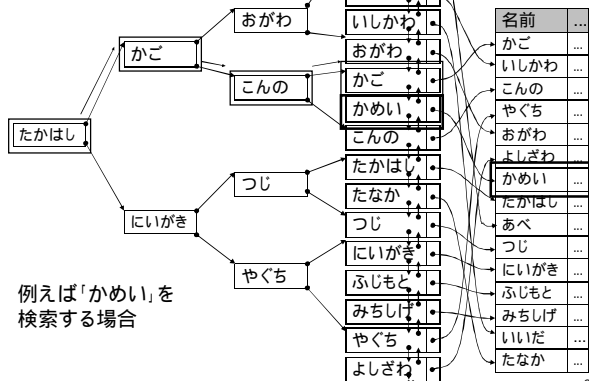


中間ノードは2つ以上のポインタ
葉ノードは連結リスト

22

B+ツリー (Balanced plus tree)

代表的なインデックスの一つ。



例えば「かめい」を
検索する場合

23

B+ツリーを使って問い合わせの処理をする手間は $O(\log n)$

テーブルの行数nが大きくなると

問い合わせの処理の効率が大きく異なる

n	log n	n
10	3	10
100	6	100
1,000	9	1,000
10,000	13	10,000
100,000	16	100,000

ただし,

インデックスを作成する手間
インデックスのためのデータ領域
が余計に必要

インデックスを作成したテーブルに
データの追加, 削除, 変更を
おこなうと
インデックスも修正する必要がある

問い合わせが中心のテーブルではインデックスは有効
(多くのDBでは, 追加, 削除, 変更は
めったにやらない場合が多い)

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

24

問い合わせの最適化 まとめ

where条件の中で単独のテーブル内で真偽が判定できる項は最初に評価する
この部分では、インデックスが利用できる場合は、さらに効率的にできる

where条件にも<SELECTリスト>にも現れない属性は最初から不要

計算途中の表の面積をなるべく小さく！

イクイジョインは、マージソートジョインなどで効率的に

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

25

障害回復

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

26

トランザクションの

一連のDB操作(参照, 更新, 追加, 削除)のうちどこかで失敗したら, それまでの一連の操作を全て破棄 (Abort) ('ロールバック (Rollback)」命令)

もとに戻すことを可能にするため, DBMSは一連の操作がコミットされるまでの操作をログ(log)に記録する

ログは, システム障害時の障害回復のためにも使われる

システムがクラッシュしても, ログが記録されていればデータベースを直近の安定状態に戻せる. ログもクラッシュすると戻らない.

ログは(企業などの大規模DBMSでは)多重化された安定ディスクに記録される.

→ RAIDディスクなど

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

27

ログの例

T1: B R₁(A), W₁(A), C
T2: B R₂(B), W₂(B),

	TransID	操作	TabID	TupID	ColID	OLD	NEW
...							
1054	T1	Begin					
1055	T1	Read	社員	007	給与		
1056	T2	Begin					
1057	T1	Write	社員	007	給与	50	100
1058	T2	Read	社員	002	給与		
1059	T2	Write	社員	002	給与	80	120
1060	T1	Commit					
...							

もしT2がRollbackされたらT2のログをもとにT2が変更した値を元に戻す

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

28

ログのしくみ

ログにはDBMSの実行しているすべてのトランザクションの操作記録(READ, WRITE, ROLLBACK, COMMIT)が保存される

DBMSは実際にデータベースに対する操作を**実行する前に**安定した記憶装置*注1にあるログに記録(この仕組みをWAL(Write-Ahead Logging) Protocolと呼ぶ)ログの記録に(システム障害などの理由で)失敗するとそれに対応する操作はデータベースには残らない

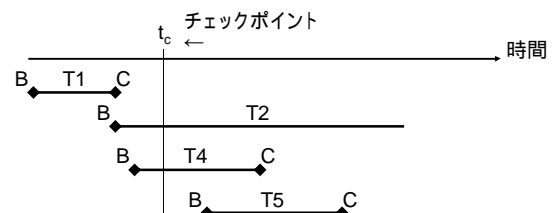
注1 (企業などの)大規模なDBMSでは, ログを記録するための(多重化され, 安定電源装置で動く)専用のHWを装備している.

同一のマシン内にログをおく場合も, ログの記録には, ファイルシステム内のファイルを使わず, 直接, ハードディスクを操作して記録する場合が多い.

(OSの(ファイルシステムなどの)機能は安全なログの記録には不十分な面がある)

29

チェックポイント法

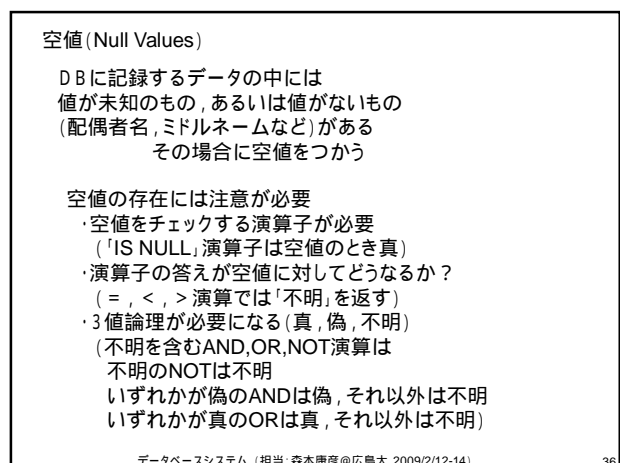
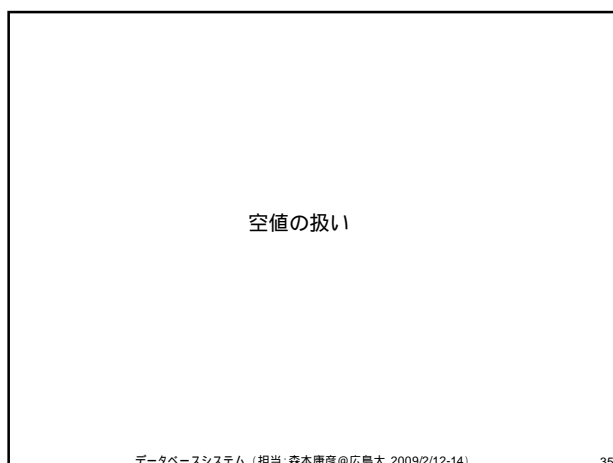
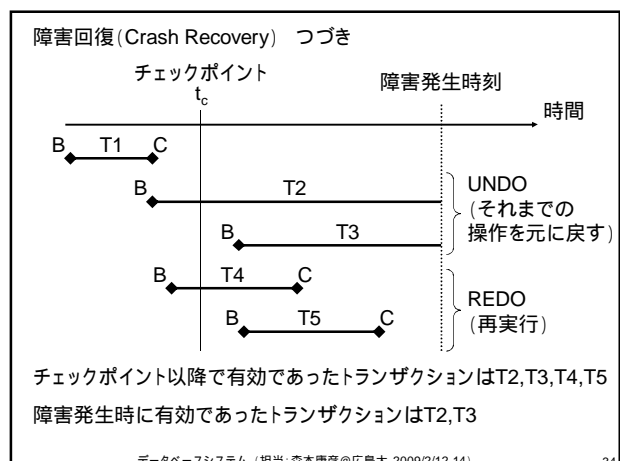
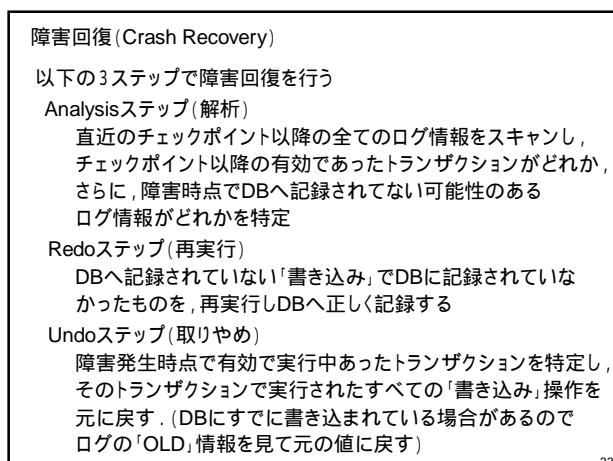
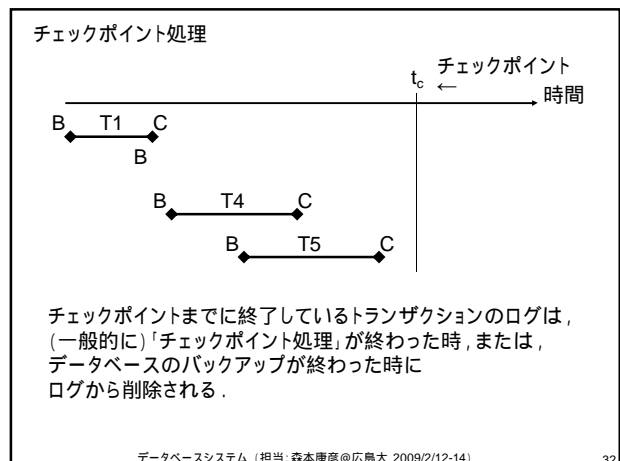
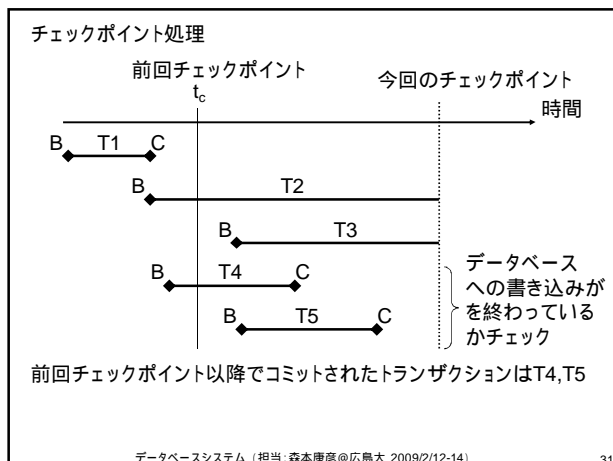


DBへの操作は, まずログに記録し, その後, DBに記録一定のタイミング(チェックポイント)でログをチェックし, チェックポイントまでに, 終了している操作がDBに反映されているかをチェック

(バッファプール上で書き換えられているだけで, 実際にハードディスクの内容が変わっていない状態もある.)

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

30



「NULL」値がある場合

mid='David' ?

customers	first	mid	last	bt	sex
	John	Fitzgerald	偽	Kennedy	M
	John	David	真	Souther	B
	Yasuhiko		不明	Morimoto	A

SELECT * FROM customers WHERE mid='David'

John	David	Souther	B	
------	-------	---------	---	--

「mid='David」はMorimotoに対しては「不明(unknown)」となる
「真(true)」でない行は選択されない。

SELECT * FROM customers WHERE mid IS NULL

Yasuhiko		Morimoto	A	M
----------	--	----------	---	---

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

「NULL」値がある場合

customers	first	mid	last	bt	sex
	John	Fitzgerald	Kennedy		M
	John	David	Souther	B	
	Yasuhiko		Morimoto	A	M

SELECT * FROM customers WHERE bt='A' and sex='M'

Yasuhiko		Morimoto	A	M
----------	--	----------	---	---

bt='A' ? sex='M' ?
不明 真
偽 不明
真 真

SELECT * FROM customers WHERE bt='A' or sex='M'

John	Fitzgerald	Kennedy		M
Yasuhiko		Morimoto	A	M

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

ビューとセキュリティ

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

DBMSの管理 (DBAが行う)

DBA (Database Administrator) の重要な仕事

- ・モデルの設計を行い、各データをリレーションにする
その際には、スキーマとIC (一貫性制約) を設定
- ・DBMSのチューニング
(HW, NWの構成とそれに応じたパラメータ設定)
- ・セキュリティの設定
(ユーザとグループ作成
ユーザーとグループに対するアクセス権限)
- ・パフォーマンスチューニング
(インデックス等の設定)

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

ビュー (View)

- ・ビューは(仮想的な)リレーション
- ・スキーマ定義のみで実際のデータ(タプル集合)を保持するものではない

CREATE VIEW YoungActiveStudents (name, grade)
AS SELECT S.name, E.grade
FROM Students S, Enrolled E
WHERE S.sid = E.sid and S.age < 21

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@ee	19	3.8

sid	cid	grade
53666	c101	C
53666	c203	B
53650	c112	A
53666	c105	B

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@ee	19	3.8

sid	cid	grade
53666	c101	C
53666	c203	B
53650	c112	A
53666	c105	B

CREATE VIEW YoungActiveStudents (name, grade)
AS SELECT S.name, E.grade
FROM Students S, Enrolled E
WHERE S.sid = E.sid and S.age < 21

YoungActiveStudents

name	grade
Jones	C
Jones	B
Jones	B
Smith	A

実際に、このテーブルは
DB内には存在しないが
存在するのように見える

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

YoungActiveStudents

name	grade
Jones	C
Jones	B
Jones	B
Smith	A

実際に、このテーブルは
DB内には存在しないが
存在するかのように見える

このビューに対する問い合わせに対して、
あたかも、そのテーブルが存在しているかのようにふるまう。

ビューはDROP VIEWコマンドで削除できる

ビューの元となるテーブルが削除された場合の
ビューの扱いは、ICの場合同様にいくつかの
対応法がある
(「元テーブルが削除されるとビューも削除」、
「ビューの元テーブルの削除を禁止」など...)

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

43

ビューとセキュリティ

ビューは必要な情報(を要約したもの)だけを
見せる手段として使われることが多い。
データをもつリレーションには、見せたくない情報を
含め多くの情報が含まれる。

```
CREATE VIEW YoungActiveStudents (name, grade)
AS SELECT S.name, E.grade
FROM Students S, Enrolled E
WHERE S.sid = E.sid and S.age < 21
```

「YoungActiveStudents」というビューをだけを見せる
場合、21歳より若い「学生」の名前と
その人の「履修」した科目の成績以外は見えない

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

44

アクセス権の設定

ユーザやDBレベル

- ・ユーザの登録
 - ・データベースの作成, 削除
- はDBAのみに与えられる

テーブル, ビューレベル

いわゆるDDL操作

- ・テーブルやビューの作成, 削除,
 - ・テーブルのスキーマの変更(列の追加, 削除)
 - ・アサーション, トリガーの設定
- は通常, DBAとDBの所有者のみに与えられている

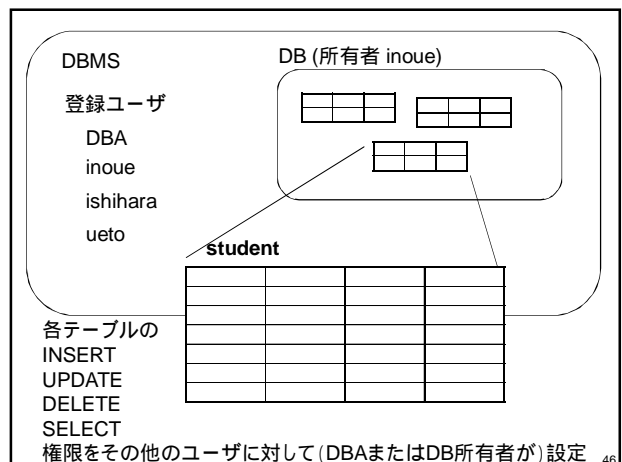
行レベルの権限

いわゆるDML操作

- ・行データの問い合わせ, 追加, 削除, 変更
- はDBAとDB所有者が, 各データベースの
一般ユーザに与える。
(一般ユーザにも権限委譲できる)

データベースシステム (担当: 森本康彦@広島大 2009/2/12-14)

45



46

テーブル, ビューに対するアクセス権の設定

行レベルのアクセス権を与える方法

Grant <Insert | Delete | Update | Select | All>
On <テーブル名 | ビュー名> 権限を与えられたユーザが
To <ユーザ名> それと同等の権限を他人に
[with Grant Option] 与えることができる

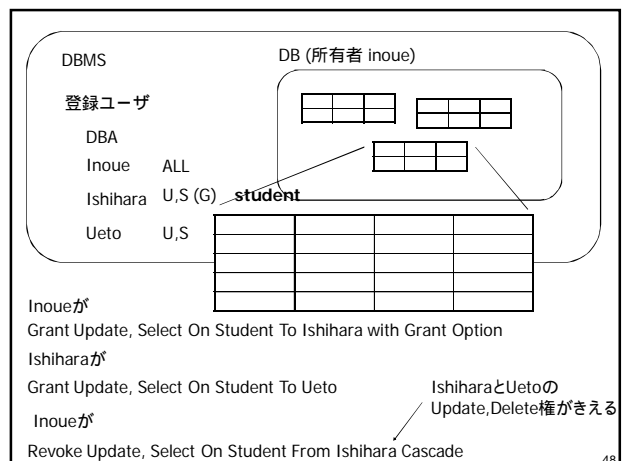
Grant Select On Student To Ishihara

アクセス権を取り消す

Revoke <Insert | Delete | Update | Select | All>
On <テーブル名 | ビュー名> 権限を剥奪されるユーザが
From <ユーザ名> 他人に与えた権限も
[Cascade] 同時に取り消す

Revoke Select On Student From Ishihara

47



48