<u>トップページ</u> -> <u>研究道具箱と教材</u> -> <u>リレーショナルデータベース入門(実践で学ぶ)</u> -> <u>テーブルの分解(table decomposition)</u> [<u>サイトマップ</u>へ] [<u>全文検索</u>へ] [<u>統計情報</u>へ]

# テーブルの分解 (table decomposition)

URL: http://www.db.is.kyushu-u.ac.jp/rinkou/addb/4.html

	name	•	teacher name	student name	score
	Databa	se	K	KK	85
table	Databa	se	K	AA	75
'results'	Databa	se	K	LL	90
	Program	ming	Α	KK	85
	Program	ming	А	LL	75
SELECT name, teacher FROM results;					
		r	name	teacher _name	
		Da	tabase	K	
		Da	tabase	K	
		Da	tabase	K	
		Prog	ramming	А	
		Prog	ramming	A	
	Quer	y res	ult is on	e new ta	ble
	nam	e	teacher _name	student _name	score
4 - 1- 1 -	nam Databa	e ase	teacher _name K	student _name KK	score 85
table	nam Databa Databa	e ise	teacher _name K K	student _name KK AA	<b>score</b> 85 75
table 'results'	nam Databa Databa Databa	e ise ise ise	teacher _name K K K	student _name KK AA LL	<b>score</b> 85 75 90
table 'results'	nam Databa Databa Databa Program	e ase ase ase ming	teacher _name K K K K A	student _name KK AA LL KK	<b>score</b> 85 75 90 85
table 'results'	nam Databa Databa Databa Program Program	e ase ase ase ming ming	teacher _name K K K A A	student _name KK AA LL KK LL	<b>score</b> 85 75 90 85 75
table 'results' SELECT DISTINCT nat FROM results;	nam Databa Databa Program Program	e ase ase ming ming	teacher _name K K K A A	student _name KK AA LL KK LL	<b>score</b> 85 75 90 85 75
table 'results' SELECT DISTINCT nat FROM results; 重複した値を持つ行は1つの Duplicate Records are Elim from the Query Pecult	name Databa Databa Databa Program Program me, tea	e ase ase ming ming achea	teacher _name K K K A A	student _name KK AA LL KK LL teacher _name	<b>score</b> 85 75 90 85 75
table 'results' SELECT DISTINCT nat FROM results; 重複した値を持つ行は1つの Duplicate Records are Elimi from the Query Result	nama Databa Databa Databa Program Program me, tea 行になる inated	e ase ase ming ming che r	teacher _name K K K A A A	student _name KK AA LL KK LL teacher _name K	<b>score</b> 85 75 90 85 75
table 'results' SELECT DISTINCT nat FROM results; 重複した値を持つ行は1つの Duplicate Records are Elimi from the Query Result	nama Databa Databa Databa Program Program me, tea	e ase ase ming ming chea r Da Prog	teacher _name K K K A A A	student _name KK AA LL KK LL teacher _name K A	<b>score</b> 85 75 90 85 75

	テー	ブルの分	解			
■ 分解前	■ 分解前 (Table Decomposition)					
results(name,	teacher_name	e, <mark>studen</mark> t	t_na	me, s	score)	
	name	teacher _name	stu _na	dent ime	score	
	Database	K	K	κ	85	
	Database	K	Α	A	75	
	Database	K	L	.L	90	
	Programming	Α	K	κ	85	
	Programming	Α	L	.L	75	
■ 分解後				4		- 
A(name, teach	ner_name), B(	name, sti	uden	it_na	me, scor	e)
<b></b>		name		stud	ent_name	score
name	teacher_name	Databa	se		KK	85
Database	ĸ	Databa	se		AA	75
Programming	A	Databa	se			90
		Program	ning		KK	85
		Program	ning		LL	70
	テー	ブルの分	<b>}</b> 解			
	(Table D	)ecompo	ositi	on)		
■ 八級 前						
■ 刀 件 们 rooulto	(nome teach	or nomo	otuc	lont	nomo o	
results		er_name,	Siuc	lent_	name, s	
■ 八級後						
A(name, teacher_name), B(name, student_name, score)					nt nom	
,	, teacher_nam	ne), B(nar	ne, s	stude	ent_name	e, score)

results の全ての属性が、分解後のテーブルに含まれていること. All attributes in 'results' must be included in the decomposed tables.

	テーブルの分解と結合								
A	A(name, teacher_name), B(name, student_name, score)								
			_	name		student	_name	S	score
	name	teacher_name		Database		Kł	<		85
	Database	K		Database		AA	۹		75
	Programming	Α		Database		Ll	_		90
				Programmin	g	Kł	<		85
				Programmin	g	Ll	-		75
SE FR Wł	SELECT A.name, A.teacher_name, B.student_name, B.score FROM A, B WHERE A.name = B.name								
紀テ	合問い合わせ	により元の Dで この		name	tea	acher_n ame	student ame	n	score
- <del>/</del>	解は無損失分	解である.		Database	K KK			85	
Ť	he original tab	le 'results' is		Database		K AA			75
С	onstructed.			Database		K	LL		90
The decomposition is		Ρ	rogramming		Α	KK		85	
Ľ	OSSLESS.		Ρ	rogramming		Α	LL		75

- テーブルの分解は、必ずしも情報無損失ではない table decompositions are not always lossless
- データベース設計には2つの異なる方針がある

Two alternatives in database design

- テーブルを情報無損失分解する
   Decompose tables losslessly
- 2. テーブルをなるべく分解しない

Do not decompose tables

 テーブルを情報無損失分解することで、データの管理 がしやすくなる場合がある

Lossless table decomposition is useful to make database management ease

・テーブル(table) <u>リレーショナルデータベース</u>での「<u>テーブル」は、多重集合(mutiset)</u>を基礎とする体系になっているということに注意 して欲しい、「テーブル」は、同じ値をもった行が複数回現れることが許される。

<sup>・</sup>SQL における**DISTINCT 指定** <u>SQL</u> では, SELECT と選択リスト内の間に「DISTINCT」と書く場合がある. これを **DISTINCT 指定**という.

DISTINCT 指定を行うと、問い合わせの結果に**重複行があるとき1つを残して他を除去**する、重複行というときは、**行** 全体が同じ値だという意味である、下記の例では、同じ注文者名が別の商品番号の注文を行った場合には重複行で はない。

SELECT DISTINCT 商品番号,注文者名 FROM 注文

DISTINCT 指定があるとき、重複行があるとき1つを残して他を除去するから、<u>SQL</u>の評価結果は多重集合にはなることは無い、必ず、リレーションになる、一方で、DISTINCT 指定がないときは、<u>SQL</u>の評価結果が多重集合になる場合がある。

補足説明しておく、下記のテーブル results を例に説明する.

name	teacher_name	student_name	score
Database	K	KK	85
Database	К	AA	75
Database	К	LL	90
Programming	Α	KK	85
Programming	Α	Ì LL	75

このテーブル results から、**属性 name と teacher\_name だけを抜き出して**新しいテーブルを作ったとする. 下記のよう に重複行が現れる. つまり新しく出来たテーブルは**多重集合**である.

name | teacher\_name Database | K Database | K Database | K Programming | A Programming | A

上記のテーブルを作るための SQL は次の通りである.

SELECT name, teacher\_name FROM results:

一方で、射影 results[name, student\_name] といったときは、属性 name と teacher\_name だけを抜き出した上で、重 **複行があるとき1つを残して他を除去**する. 結果は次の通りである.

name | teacher\_name Database | K Programming | A

射影を行うための SQL は、次のように DISTINCT 指定が付く.

SELECT **DISTINCT** name, teacher\_name FROM results;

#### ・テーブルの分解

ある<u>テーブル</u>のテーブル名が R, 属性名が A1, A2, …, An であるとする. また, このテーブルには **重複行が無い**もの と仮定する. このテーブルの, m 個のテーブル X1, X2, ..., Xm への分解は次のように定義される.

X1, X2, …, Xm をRの全属性集合{A1, A2,…, An}の部分集合で,

 $X1 \cup X2 \cup \cdots \cup Xm = \{A1, A2, \cdots, An\}$ 

なる条件を満たすとき(XiとXiは共通集合をもってよい), <u>テーブル</u>を m 個の射影 R[X1], R[X2], …, R[Xm] で置き換えるこ とを, このテーブルの分解という.

※ 重複行を持つようなテーブルについては、分解を考えないことにする(重複行を持つテーブルについては、無損失 分解であることの定義が「難しい」から).

## · 単純値 atomic value

分解不可能な値のこと. 直積集合の要素や, べき集合の要素は単純値ではないと考えるのが普通である.

- 。直積集合の例:属性名「family name」のドメインと,属性名「given name」のドメインの直積集合
- ベき集合の例: ドメイン {みかん, りんご, バナナ}から構成されるべき集合は, {φ, {みかん}, {りんご}, {バナナ}, ナ), {みかん, りんご}, {りんご, バナナ}, {バナナ, みかん}, {みかん, りんご, バナナ})

## 演習

## 演習で行うこと

- ・<u>Sqliteman の起動と終了</u> (Start Sqliteman)
- ・<u>Sqliteman で既存のデータベースを開く</u>(Open an existing database using Sqliteman)
- ・ <u>SQL を用いたテーブル定義と一貫性制約の記述</u> (Table defintion and integrity constraint specification using SQL) リレーショナル・スキーマ (relational schema): **results(name, teacher\_name, student\_name, score)**

SQL プログラム:

CREATE TABLE **results** ( name TEXT NOT NULL, teacher\_name TEXT NOT NULL, student\_name TEXT NOT NULL, INTEGER NOT NULL CHECK ( score >= 0 AND score <= 100 ), score UNIQUE (name, student\_name) UNIQUE (teacher\_name, student\_name) ); <u>SQL を用いたテーブルへの行の挿入</u> (Insert rows into a table using SQL) BEGIN TRANSACTION: INSERT INTO results VALUES( Database , K', KK', 85); K', AA', 75); K', LL', 90); INSERT INTO results VALUES( Database, K, AA, 75); INSERT INTO results VALUES( Database, K, AA, 75); INSERT INTO results VALUES( Database, K, LL, 90); INSERT INTO results VALUES( Programming, A, KK, 85); INSERT INTO results VALUES( Programming, A, LL, 75); COMMIT: · <u>Sqliteman を用いたデータのブラウズ</u> (Browse Data using Sqliteman) <u>DISTINCT を含む SQL の例</u> (SQL with DISTINCT) SELECT **DISTINCT** name, teacher\_name FROM results: SELECT name, teacher\_name FROM results: SELECT **DISTINCT** score FROM results SELECT score FROM results <u>SQL を用いたテーブルの分解</u>(table decomposition using SQL) **CREATE TABLE A** AS SELECT DISTINCT name, teacher\_name FROM results: **CREATE TABLE B** AS SELECT DISTINCT name, student\_name, score FROM results ・<u>結合問い合わせ</u> (join query) SELECT A.name, A.teacher\_name, B.student\_name, B.score FROM A B WHERE A. name = B. name: 情報無損失分解にならない分解(Inspect lossless decomposition) **CREATE TABLE C** AS SELECT DISTINCT name, student\_name FROM results: **CREATE TABLE D** AS SELECT DISTINCT teacher\_name, student\_name, score FROM results: SELECT C.name, D.teacher\_name, C.student\_name, D.score FROM C, D WHERE C.student\_name = D.student\_name SQLite の SQL 演習に関連する部分

SQLite の SQL の説明は <u>http://www.hwaci.com/sw/sqlite/lang.html</u> (English Web Page) にある.

· SELECT DISTINCT ...

同一の値を持つ行が複数あるとき、1つを残して除去する

CREATE TABLE AS <table-name> SELECT ...

SQL 問い合わせ結果から新しいテーブルを作る

## Sqliteman の起動と終了 (Start Sqliteman)

1. Sqliteman の起動 (Start Sqliteman)

## ■ Ubuntu での SQLiteman の起動例

「プログラミング」→「Sqliteman」と操作する.



SQLiteman の新しいウインドウが開く(A New window appears)

<mark>Eile ⊆</mark> ontext <u>D</u> atabase <u>S</u> ystem <u>H</u> elp	
Eile       Context       Database       System       Help         Schema       Pragmas         Database       Database	Image: Second
Windows での SQLiteman の起動例	

「Sqliteman」のアイコンをダブルクリック (double click "Sqliteman.exe")



Sqliteman のウインドウが開く(A New window appears)

e <u>C</u> ontext Database System <u>H</u> elp	
Schems Pragmas	
Database	
	Full Wern Term Wern Script Output

# Sqliteman で既存のデータベースを開く(Open an existing database using Sqliteman) すでに作成済みのデータベースを、下記の手順で開くことができる。

以下の手順で, 既存のデータベースファイルを開く. (Open an existing database file)

1. 「File」→「Open」

Sqliteman		
<u>File Context Dat</u>	abase <u>S</u> ystem <u>H</u> elp	
<u>N</u> ew	Ctrl+N	
0pen	Ctrl+O	
Recent Databa	363	1
<u>P</u> references		
🖾 E <u>x</u> it	Ctrl+Q	
		Col: 1 Row: 1/1

## 2. データベースファイルを開く(Open Database File)

■ Ubuntu での実行例(「SQLite/mydb」を開く場合)

データベースファイル **SQLite/mydb** を選び,「開く」をクリック (Click '開く' after choosing the database file "SQLite /mydb")

🔛 🗔 Open Database					
🖉 🖵 home 🗟 wind	dowslike SQLite				
場所( <u>P</u> )	名前	*	サイズ	最終変更日	
🔍 検索	📄 mydb		0 パイト	17:18	
🕗 最近開いたファイル	🗇 sqlite3_analyzer		1.7 MB	2010年08月24日	
🗟 windowslike					
□ デスクトップ					
ファイル・システム					
フロッピー・ドライブ					
■ ドキュメント					
■ 首衆				=	
■ ■ ■ ■ ■					
🗟 ダウンロード					
				-	
◆追加( <u>A</u> ) ── 削除( <u>R</u> )			S	QLite database 🔻	
		8±+	ンセルに		١
			- CIVIC		,

■ Windows での実行例(「C:¥SQLite¥mydb」を開く場合)

データベースファイル **C:¥SQLite¥mydb** を選び,「開く」をクリック (Click '開く' after choosing the database file "C:¥SQLite¥mydb")

Open Database					<u>? ×</u>
ファイルの場所の	SQLite		•	🗕 🗈 💣 🎟	•
していたファイル	nydb ■sqlite3.exe				
で デスクトップ					
ک ۲۲ (۲۴ אר)					
پر ایک					
र्ग रूग-७					
	ファイル名( <u>N</u> ):			•	
	ファイルの種類(1):	SQLite database (*)		<u> </u>	**>セル

要するに、/home/<ユーザ名>/SQLite の下の mydb を選ぶ.

SQL を用いたテーブル定義と一 貢性制約の記述 (Table definition and integrity constraint specification using SQL) SQL を用いて, results テーブルを定義し、一貫性制約を記述する. (Define 'results' table and specify integrity constrants of the table using SQL)

リレーショナル・スキーマ (relational schema): results( name, teacher\_name, student\_name, score )

1. results テーブルの定義 (Define a table)

次の SQL を入力し、「Run SQL」のアイコンをクリック (Write the following SQL, and click "Run SQL" icon).

CREATE TABLE **results** ( name TEXT NOT NULL, teacher\_name TEXT NOT NULL, student\_name TEXT NOT NULL, score INTEGER NOT NULL CHECK (score >= 0 AND score <= 100), UNIQUE (name, student\_name), UNIQUE (teacher\_name, student\_name) );

※「SQL Editor」のウインドウには、SQL プログラムを書くことができる. In the '**SQL string**' window, you can write down **SQL program(s)**.

S mydb – Sqliteman	
<u>File Context</u> <u>Database</u> <u>Syst</u>	em <u>H</u> elp
Sche <u>m</u> a <u>P</u> ragmas	
Database → → main → → Tables (6) → - bundles → order_rec··· → products → requests	1       ECREATE TABLE results (         2       name       TEXT NOT NULL,         3       teacher_name TEXT NOT NULL,         4       student_name TEXT NOT NULL,         5       score         INTEGER NOT NULL CHECK (score >= 0 AND score <= 100 ),

2. コンソールの確認 (Inspect console)

```
エラーメッセージが出ていないことを確認
```

Query OK Row(s) returned: 0 (More rows can be fetched. Scroll the resultset for more rows and/or read the documentation.) CREATE TABLE results ( name TEXT NOT NULL, teacher\_name TEXT NOT NULL, student\_name TEXT NOT NULL, score INTEGER NOT NULL CHECK (score >= 0 AND score <= 100), UNIQUE (name, student\_name), UNIQUE (teacher\_name, student\_name));

## SQL を用いたテーブルへの行の挿入 (Insert rows into a table using SQL)

次のような results テーブルを作る. (Construct table 'results')

name	teacher_name		student_name	score
Database	К		KK	85
Database	K	1	AA	75
Database	K	Î.	LL	90
Programming	Α	Í.	KK	85
Programming	A	Î.	LL	75

以下の手順で, SQL を用いて results テーブルへの行の挿入を行う (Insert rows into table 'results' using SQL)

#### 1. SQL プログラムの記述

「INSERT INTO ...」は行の挿入. ここには **5つの SQL 文**を書き,「BEGIN TRANSACTION」と「COMMIT」で囲む. ("INSERT INTO ..." means inserting a row into a table. Four SQL statements are wrote).

BEGIN TRANSACTION: INSERT INTO results VALUES( Database', K', KK', 85); INSERT INTO results VALUES( Database', K', AA', 75); INSERT INTO results VALUES( Database', K', LL', 90); INSERT INTO results VALUES( Programming', A', KK', 85); INSERT INTO results VALUES( Programming', A', LL', 75); COMMIT:

S mydb – Sqliteman	
<u>File Context Database System H</u> elp	
Sche <u>m</u> a <u>P</u> ragmas	🕨 🦹 🖿 🔚 🖾 🏝 🔍
Database	1       BEGIN TRANSACTION;         2       INSERT INTO results VALUES('Database', 'K', 'KK', 85 );         3       INSERT INTO results VALUES('Database', 'K', 'AA', 75 );         4       INSERT INTO results VALUES('Database', 'K', 'LL', 90 );         5       INSERT INTO results VALUES('Programming', 'A', 'KK', 85 );         6       INSERT INTO results VALUES('Programming', 'A', 'LL', 75 );         7       COMMIT;

2. 複数の SQL 文の一括実行 (Run multiple SQL statements)

複数の SQL 文を一括実行したいので, **カーソルを先頭行に移動**した後に、「Run multiple SQL statements ...」のボタ ンをクリックする. 「Move the cursor to the top statement. Click "Run multiple SQL statements from current cursor position in one batch" icon)



3. 「Script Output」ウインドウの確認 (Inspect "Script Output" window)

エラーメッセージが出ていないことを確認

Full V	/iew Item View Script Output		
1	Script started		
2	FI BEGIN TRANSACTION		
3	No error		
4			
5	INSERT INTO results VALUES ('Database', 'K',	'KK', 85.);	
6	No error		
17			
8	INSERT INTO results VALUES ('Database', 'K',	'AA', 75 );	
9	No error		
10			
11	INSERT INTO results VALUES ('Database', 'K',	'LL', 90 );	
12	No error		
13			
14	INSERT INTO results VALUES ('Programming',	'A', 'KK', 85 );	
15	No error		
16	-		
17	INSERT INTO results VALUES ('Programming',	(A', 'LL', 75 );	
18	No error		
19			
20	COMMIT		
21	No error		
22			
23	Script finished		
24			

# Sqliteman を用いたデータのブラウズ (Browse Data using Sqliteman)

## ・results テーブル

まず、オブジェクト・ブラウザ (Object Browser) の中の「**Tables**」を展開 (Click 'Tables')



次に, テーブル resultsを選ぶ (Select table 'results')

🥿 mydh – Saliten	an
File Context Data	abase System Help
Sche <u>m</u> a <u>P</u> ragi	mas
Database	
<ul> <li>□ main</li> <li>□ main</li> <li>□ Tabl</li> <li>□ produ</li> <l< td=""><th>es (7) les records acts acts acts acts acts acts acts act</th></l<></ul>	es (7) les records acts acts acts acts acts acts acts act

テーブル **results**が表示される (table 'results' appears)

Full View Item View Script Output					
name	teacher_name	student_name	score		
1 Database	К	КК	85		
2 Database	К	AA	75		
3 Database	К	LL	90		
4 Programming	A	КК	85		
5 Programming	A	LL	75		

※ もし, データに間違いがあれば, このウインドウで修正できる (If you find any mistakes, you can modify the data using this window).

# DISTINCT を含む SQL の例 (SQL with DISTINCT)

# DISTINCT を付ける (with DISTINCT)

SELECT	DISTINCT	name,	teacher_name
FROM re	esults:		

DR 🕨 📼 🗉 🖆 🏝 🔍	
1 2 SELECT DISTINCT name, teacher name FROM results	
•	•
Duration: 0.016 seconds	* Col: 14 Row: 2/2
] 26 21 21 27 27 27 27 27 27 27 27 27 27 27 27 27	
Full View Item View Script Output	
name	teacher_name
1 Database	К
2 Programming	Α

## DISTINCT を付けない (without DISTINCT)

SELECT name, teacher_name FROM results:						
1     SELECT name, teacher_name       2     FROM results;						
	Þ					
Duration: 0.015 seconds	Duration: 0.015 seconds Total Row: 2/2					
]B B B B & Ø   ■ D   □						
Full View I Item View Script Output						
name	teacher_name					
1 Database	к					
2 Database	К					
3 Database	ĸ					
4 Programming	A					
5 Programming	A					

## DISTINCT を付ける (with DISTINCT)

SELECT **DISTINCT** score FROM results:

🕟 🧣 🕪 📼 📄 🗁 🏝 🔍	
1 SELECT DISTINCT score 2 FROM results;	
	Þ
Duration: 0 seconds	* Col: 14 Row: 2/2
] B B B   S S   M 🕑 ] 📭	
Full View Item View Script Output	
score	
1	75
2	85
3	90

DISTINCT を付けない (without DISTINCT)

SELECT score FROM results:	
<b>                                    </b>	
1 SELECT score 2 FROM results;	
	F
Duration: 0 seconds	* Col: 14 Row: 2/2
Full View Item View Script Output	
score	
1	85
2	75
3	90
4	85
5	75

## SQL を用いたテーブルの分解 (table decomposition using SQL)

results( name, teacher\_name, student\_name, score )を 2つのテーブル

- A(name, teacher\_name)
- $\cdot$  B(name, student\_score, score)

に分解. テーブルを分解するときは重複行を除去するという規則があることに注意. DISTINCTを使う.

'results' table is decomposed into two tables A and B. Duplicate rows in A and B are eliminated when decomposition (because of the definition of "table decomposition"). Use 'DISTINCT'.

SQL の実行 (Execute SQL)

<b>CREAT</b> Selec From	<b>E TABLE A</b> AS T DISTINCT <b>name</b> , <b>teacher_name</b> results:
	🤝 mydb - Sqliteman
	<u>File Context Database System Help</u>
	Sche <u>m</u> a Pragmas
	Database     1     CREATE TABLE A AS       Imain     2     SELECT DISTINCT name, teacher_name       Imain     3     FROM results;

データのブラウズ機能を使いテーブル A を確認 (Browse Table A)

Full View	Item View	Script Output		
		name	teacher_name	١
1 Databas	se		ĸ	I
2 Program	nming		A	l
				′

SQL の実行 (Execute SQL)

### **CREATE TABLE B** AS

SELECT DISTINCT **name**. **student\_name**. **score** FROM results:

S mydb – Sqliteman	
<u>File Context Database System</u>	<u>l</u> elp
Sche <u>m</u> a <u>P</u> ragmas	
Database	CREATE TABLE B AS     SELECT DISTINCT name, student_name, score
🔁 🛅 Tables (8)	3 FROM results

データのブラウズ機能を使いテーブル B を確認 (Browse Table B)

Full View Item View Script Output						
name	student_name	score				
1 Database	AA	75				
2 Database	КК	85				
3 Database	LL	90				
4 Programming	КК	85				
5 Programming	LL	75				
	1					

## 結合問い合わせ (join query)

**元のテーブル results が, 分解後の2つのテーブル A, B から復元できることを確認**しておこう. テーブル results は, テーブル A と B から復元できるから, データベース設計として.

- ・テーブル results をデータベースに格納するか、あるいは、
- ・テーブル results でなく、テーブルAとBをデータベースに格納する

という 2つの案がありえることになる.(では、どちらが良いのか? 今度の授業で明らかにしていく)

Examine that the original table 'results' can be constructed from the two tables **A** and **B**. It means that there are two alternatives in database design.

- 1. Store 'results' into database, or
- 2. Store 'A' and 'B' instead of 'results' in database.

SELECT A. name,	A.teacher_name,	B.student_name,	B.score
FROM A, B			
WHERE A. name =	B. name ;		

	) 🕅 🖿 📄 🔛	2 2 S		
1 2 3	SELECT Aname, Ateacher_na FROM A, B WHERE Aname = Bname;	me, B.student_name, B.score		
				•
Duratio	on: O seconds			* Col: 23 Row: 3/3
Full	View   Item View   Script Ou	itput		
	name	teacher_name	student_name	score
1	Database	К	AA	75
2	Database	К	КК	85
3	Database	К	LL	90
4	Programming	A	КК	85
5	Programming	A	LL	75

## 情報無損失分解にならない分解 (Inspect lossless decomposition)

今度は、テーブルの分解の仕方を変える、分解後のテーブルから、もとのテーブルに戻らない場合がある、このことを確認しておく、

Table decompositions are not always LOSSLESS.

SQL の実行 (Execute SQL)

CREATE TABLE C AS SELECT DISTINCT name, student_name FROM results:	
🔄 mydb - Sqliteman	
<u>File Context</u> Database System	<u>H</u> elp
Sche <u>m</u> a <u>P</u> ragmas	
Database ⊡- ⊜ main ⊕- Ⅲ Tables (9)	1     CREATE TABLE C AS       2     SELECT DISTINCT name, student_name       3     FROM results

データのブラウズ (Browse Data)

F	III View   Item View   Script Output	
ſſ	name	student_name
1	Database	AA
2	Database	КК
	Database	LL
7	Programming	КК
Ī	Programming	LL
Y		· · · · · · · · · · · · · · · · · · ·

SQL の実行 (Execute SQL)

CREATE TABLE D AS SELECT DISTINCT teacher\_name, student\_name, score FROM results:

S mydb - Sqliteman	
<u>File Context</u> <u>D</u> atabase <u>S</u> ystem	<u>H</u> elp
Sche <u>m</u> a <u>P</u> ragmas	
Database ⊡ 😖 main ⊡ 🎫 Tables (10)	1         CREATE TABLE D AS           2         SELECT DISTINCT teacher_name, student_name, score           3         FROM results;

データのブラウズ (Browse Data)

Full View Item View Script Output		
teacher_name	student_name	score
1 A	кк	8
2 A	LL	75
3 K	AA	75
4 K	кк	85
5 K	LL	90
	1	1

## テーブル C, D からは, テーブル results を構築できない (results' table can not be constructed from C and D).

SELECT C.name, D.teacher\_name, C.student\_name, D.score FROM C, D WHERE C.student\_name = D.student\_name:

SELECT C.name, D.t FROM C, D WHERE C.student_n	eacher_name, C.student_name, D.score ame = D.student_name;		
ration: 0.016 seconds			* Col: 39 Row:
	* 🛛 💽 🖉 👼		
Full View   Item View	Script Output		
Full View Item View	Script Output   teacher_name	student_name	score
Full View Item View name	Script Output teacher_name	student_name	score 75
Full View Item View name Database Database	Script Output teacher_name K A	Student_name AA KK	score 75 85
Full View Item View name 1 Database 2 Database 3 Database	Script Output teacher_name K A K K	AA KK KK	score 75 85 85
Full View Item View name 1 Database 2 Database 3 Database 4 Database	Script Output teacher_name K A K A A A A	AA KK KK LL	score 75 85 85 75
Full View Item View name 1 Database 2 Database 3 Database 4 Database 5 Database	Script Output teacher_name K A K A K K K K K K K	AA KK KK LL LL	score 75 75 85 85 75 90
Full View Item View name 1 Database 2 Database 3 Database 4 Database 5 Database 6 Programming	Script Output teacher_name K A K A K A K A A A A A A A A A A A A	AA KK KK LL LL KK	score 75 85 85 75 90 85
Full View Item View name Database Database Database Database Database Database Database Programming Programming	Script Output teacher_name K A K A K A K A K A K K K K K K K K K	AA AA KK KK LL LL LL KK KK	score 75 85 85 75 90 85 85
Full View Item View name Database Database Database Database Database Database Programming Programming Programming	Script Output teacher_name K A K A K A K A K A K A A K A A A A A	AA KK KK LL LL KK KK KK LL	score 75 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5

# 演習問題と解答例

次の問いに答えよ. その後, 下記の解答例を確認せよ. Answer the following questions. Then, inspect answers described below.

問い (Questions)

1. 次の PTABLE テーブルに関する問題 (About the following 'PTABLE' table)

name	type	color
apple apple rose	fruit   fruit   flower	red blue white
rose	flower	yellow

1. 次の SQL の評価結果は何か? What is the result of the following SQL?

SELECT **DISTINCT** name, type FROM PTABLE:

2. 次の SQL の評価結果は何か? What is the result of the following SQL?

SELECT DISTINCT name FROM PTABLE:

解答例 (Answers)

※ 問い合わせ結果は1つのテーブルになる. その属性名には, 元のテーブル名と属性名をドットでつなげたドット記法を用いている.

1. SELECT DISTINCT name, type FROM PTABLE;

name	type
apple	fruit
rose	flower

2. SELECT DISTINCT name FROM PTABLE:

name
apple
1036