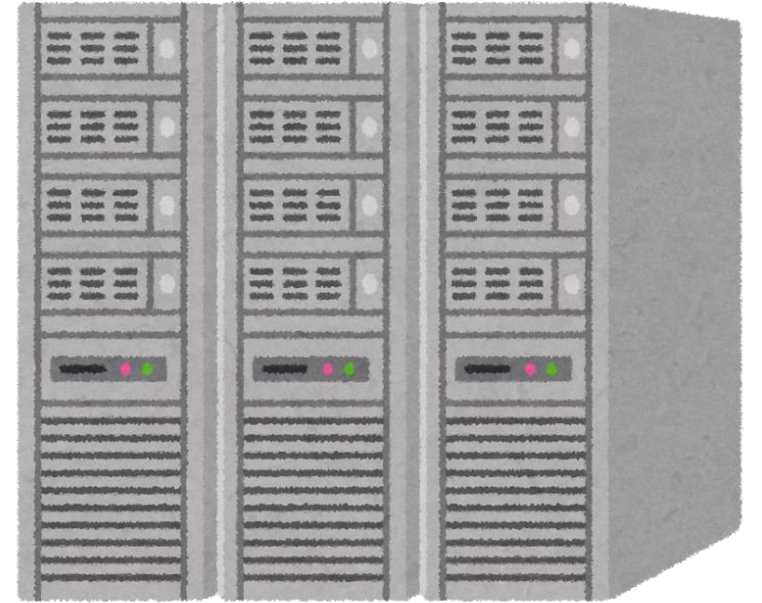
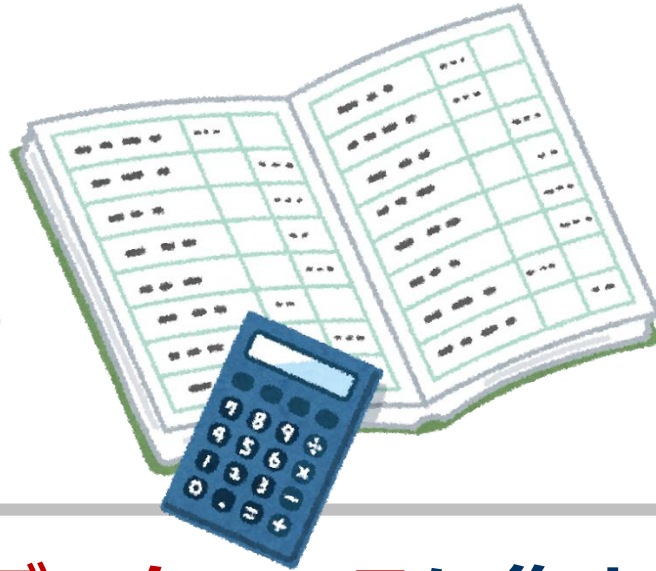


データベースシステム入門

6. SQL の select from where, 結合, 結合条件,
SQL で結合を行う

今日学習することはなぜ大切なのか



リレーショナルデータベースに集めた生データを使い、いろいろな分析を行ってデータ活用

- ◆ SQL の主要機能の総復習

select, from, where

- ◆ 「**結合**」についてより詳しく説明

6-1 SQL の select, from, where

リレーショナルデータベースシステム（復習）

コンピュータ



リレーショナル
データベース
管理システム



記憶
装置

リレーショナル
データベース

あわせて
リレーショナルデータベースシステム

	id	name	teacher_name	student_name	score	created_at	updated_at
1	1	Database	K	KK	85	2014-09-13 00:46:17	{null}
2	2	Database	K	AA	75	2014-09-13 00:48:26	{null}
3	3	Database	K	LL	90	2014-09-13 00:48:26	{null}
4	4	Programming	A	KK	85	2014-09-13 00:48:26	{null}
5	5	Programming	A	LL	75	2014-09-13 00:48:26	{null}

	id	year	month	day	customer_name	product_id	qty	created_at
1	1001	2014	10	26	kaneko	1	10	2014-09-14 14:28:38
2	1002	2014	10	26	miyamoto	2	2	2014-09-14 14:28:38
3	1003	2014	10	27	kaneko	3	8	2014-09-14 14:28:38
4	1004	2014	10	27	kaneko	3	8	2014-09-14 14:28:38

データの種類ごとに分かれた、
たくさんの**テーブル**が格納される

SQLをマスターするには

◆ SQL のキーワード

create table テーブル定義

select	射影 など
from	扱うテーブルの指定
where	選択, 結合 など

max 最大

min 最小

count 数え上げ

group by 集約（グループの基準の指定）

order by ソート（並べ替え）

distinct 重複除去

など

◆ SQL のデータ型

短いテキスト char

長いテキスト text

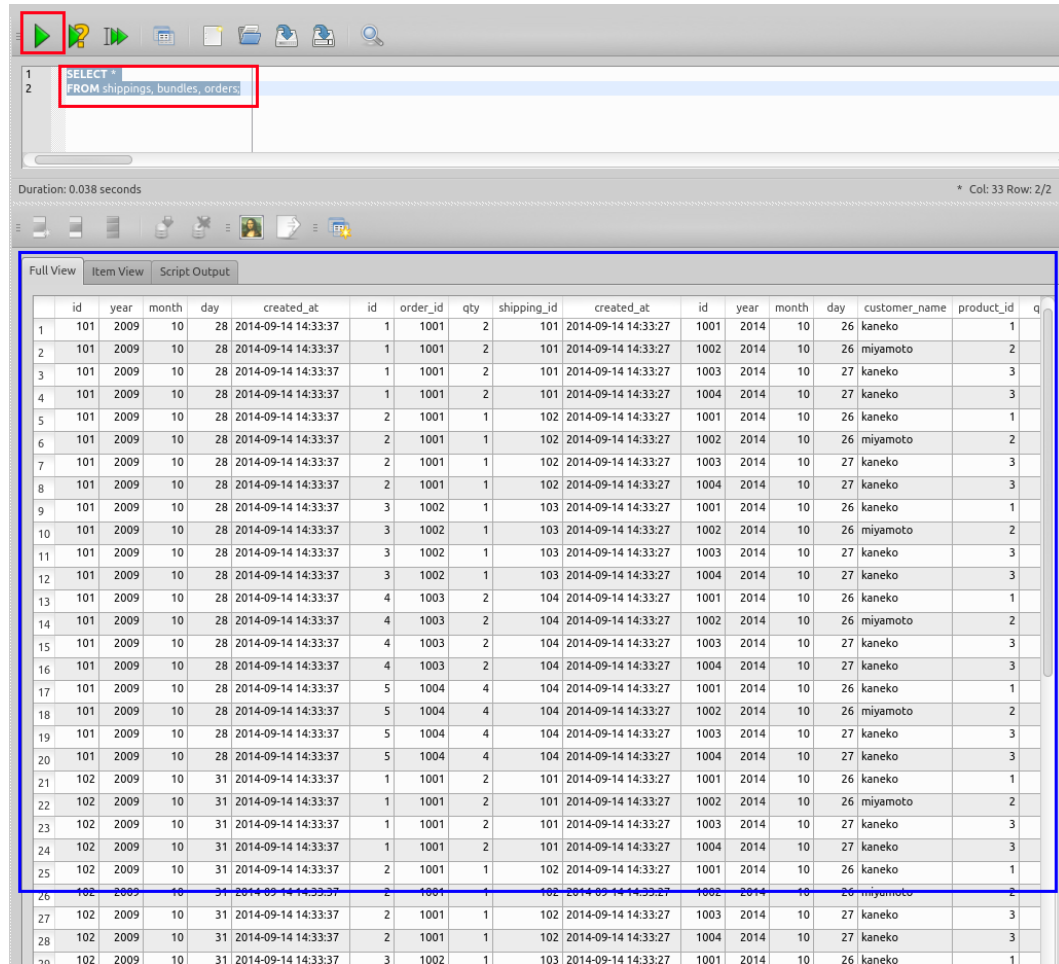
数値 integer, real

日付／時刻 datetime

YesやNo bit

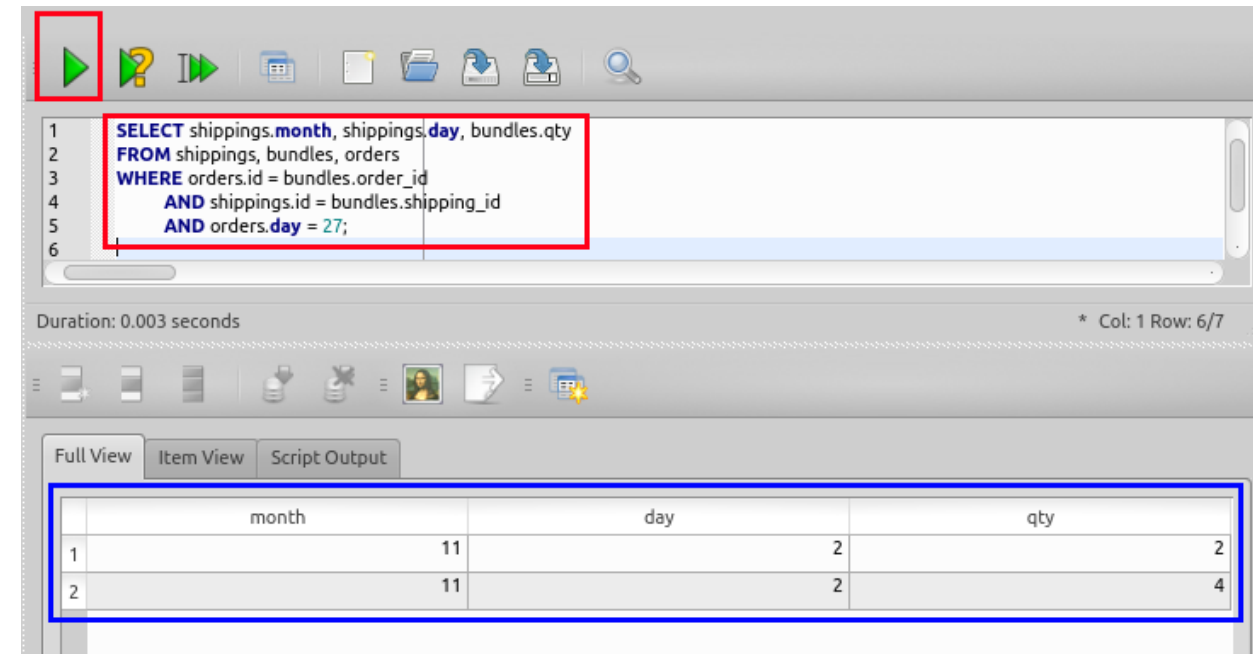
など

欲しいデータを探す，複数のテーブルを1つに結合する = SQL でできる



The screenshot shows a database tool interface. At the top, a SQL query is entered in a text area: `SELECT * FROM shippings, bundles, orders;`. Below the query, the execution duration is 0.038 seconds. The results are displayed in a table with 25 rows and 14 columns. The columns are: id, year, month, day, created_at, id, order_id, qty, shipping_id, created_at, id, year, month, day, customer_name, product_id. The data represents a list of orders with their details.

	id	year	month	day	created_at	id	order_id	qty	shipping_id	created_at	id	year	month	day	customer_name	product_id
1	101	2009	10	28	2014-09-14 14:33:37	1	1001	2	101	2014-09-14 14:33:27	1001	2014	10	26	kaneko	1
2	101	2009	10	28	2014-09-14 14:33:37	1	1001	2	101	2014-09-14 14:33:27	1002	2014	10	26	miyamoto	2
3	101	2009	10	28	2014-09-14 14:33:37	1	1001	2	101	2014-09-14 14:33:27	1003	2014	10	27	kaneko	3
4	101	2009	10	28	2014-09-14 14:33:37	1	1001	2	101	2014-09-14 14:33:27	1004	2014	10	27	kaneko	3
5	101	2009	10	28	2014-09-14 14:33:37	2	1001	1	102	2014-09-14 14:33:27	1001	2014	10	26	kaneko	1
6	101	2009	10	28	2014-09-14 14:33:37	2	1001	1	102	2014-09-14 14:33:27	1002	2014	10	26	miyamoto	2
7	101	2009	10	28	2014-09-14 14:33:37	2	1001	1	102	2014-09-14 14:33:27	1003	2014	10	27	kaneko	3
8	101	2009	10	28	2014-09-14 14:33:37	2	1001	1	102	2014-09-14 14:33:27	1004	2014	10	27	kaneko	3
9	101	2009	10	28	2014-09-14 14:33:37	3	1002	1	103	2014-09-14 14:33:27	1001	2014	10	26	kaneko	1
10	101	2009	10	28	2014-09-14 14:33:37	3	1002	1	103	2014-09-14 14:33:27	1002	2014	10	26	miyamoto	2
11	101	2009	10	28	2014-09-14 14:33:37	3	1002	1	103	2014-09-14 14:33:27	1003	2014	10	27	kaneko	3
12	101	2009	10	28	2014-09-14 14:33:37	3	1002	1	103	2014-09-14 14:33:27	1004	2014	10	27	kaneko	3
13	101	2009	10	28	2014-09-14 14:33:37	4	1003	2	104	2014-09-14 14:33:27	1001	2014	10	26	kaneko	1
14	101	2009	10	28	2014-09-14 14:33:37	4	1003	2	104	2014-09-14 14:33:27	1002	2014	10	26	miyamoto	2
15	101	2009	10	28	2014-09-14 14:33:37	4	1003	2	104	2014-09-14 14:33:27	1003	2014	10	27	kaneko	3
16	101	2009	10	28	2014-09-14 14:33:37	4	1003	2	104	2014-09-14 14:33:27	1004	2014	10	27	kaneko	3
17	101	2009	10	28	2014-09-14 14:33:37	5	1004	4	104	2014-09-14 14:33:27	1001	2014	10	26	kaneko	1
18	101	2009	10	28	2014-09-14 14:33:37	5	1004	4	104	2014-09-14 14:33:27	1002	2014	10	26	miyamoto	2
19	101	2009	10	28	2014-09-14 14:33:37	5	1004	4	104	2014-09-14 14:33:27	1003	2014	10	27	kaneko	3
20	101	2009	10	28	2014-09-14 14:33:37	5	1004	4	104	2014-09-14 14:33:27	1004	2014	10	27	kaneko	3
21	102	2009	10	31	2014-09-14 14:33:37	1	1001	2	101	2014-09-14 14:33:27	1001	2014	10	26	kaneko	1
22	102	2009	10	31	2014-09-14 14:33:37	1	1001	2	101	2014-09-14 14:33:27	1002	2014	10	26	miyamoto	2
23	102	2009	10	31	2014-09-14 14:33:37	1	1001	2	101	2014-09-14 14:33:27	1003	2014	10	27	kaneko	3
24	102	2009	10	31	2014-09-14 14:33:37	1	1001	2	101	2014-09-14 14:33:27	1004	2014	10	27	kaneko	3
25	102	2009	10	31	2014-09-14 14:33:37	2	1001	1	102	2014-09-14 14:33:27	1001	2014	10	26	kaneko	1
26	102	2009	10	31	2014-09-14 14:33:37	2	1001	1	102	2014-09-14 14:33:27	1002	2014	10	26	miyamoto	2
27	102	2009	10	31	2014-09-14 14:33:37	2	1001	1	102	2014-09-14 14:33:27	1003	2014	10	27	kaneko	3
28	102	2009	10	31	2014-09-14 14:33:37	2	1001	1	102	2014-09-14 14:33:27	1004	2014	10	27	kaneko	3
29	102	2009	10	31	2014-09-14 14:33:37	3	1002	1	103	2014-09-14 14:33:27	1001	2014	10	26	kaneko	1



The screenshot shows a database tool interface. At the top, a SQL query is entered in a text area: `SELECT shippings.month, shippings.day, bundles.qty FROM shippings, bundles, orders WHERE orders.id = bundles.order_id AND shippings.id = bundles.shipping_id AND orders.day = 27;`. Below the query, the execution duration is 0.003 seconds. The results are displayed in a table with 2 rows and 3 columns. The columns are: month, day, qty. The data represents the filtered results of the query.

	month	day	qty
1	11	2	2
2	11	2	4

SQLはコマンド言語

1つのテーブルを複数のテーブルに分解したい、 ということもある = SQL でできる

テーブルの分解 (Table Decomposition)

■ 分解前

results(name, teacher_name, student_name, score)

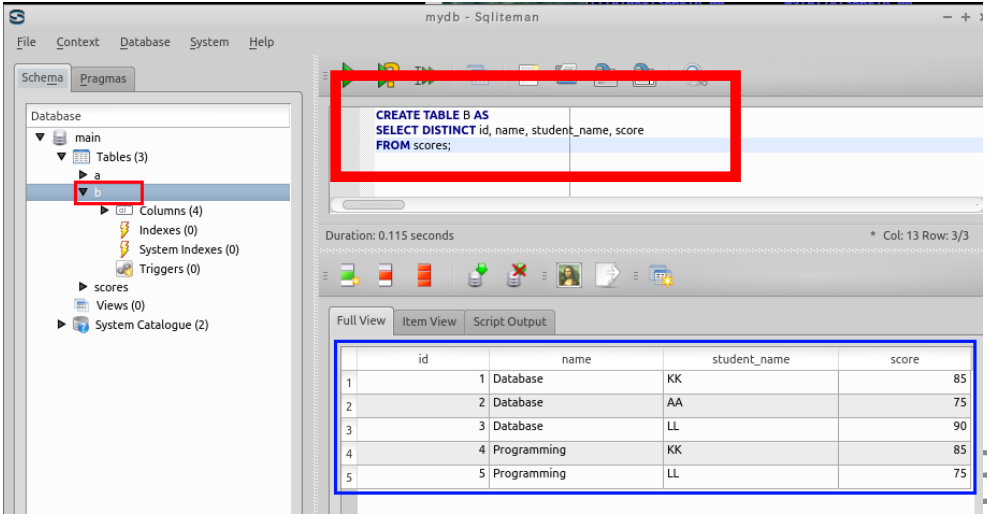
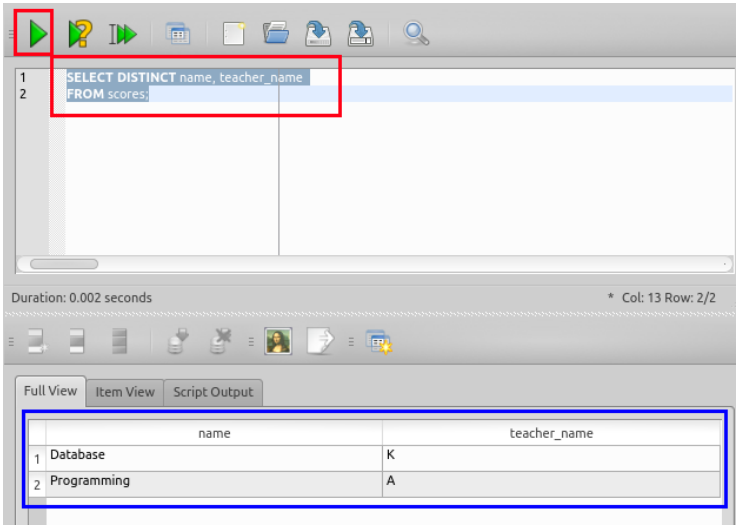
name	teacher_name	student_name	score
Database	K	KK	85
Database	K	AA	75
Database	K	LL	90
Programming	A	KK	85
Programming	A	LL	75

■ 分解後

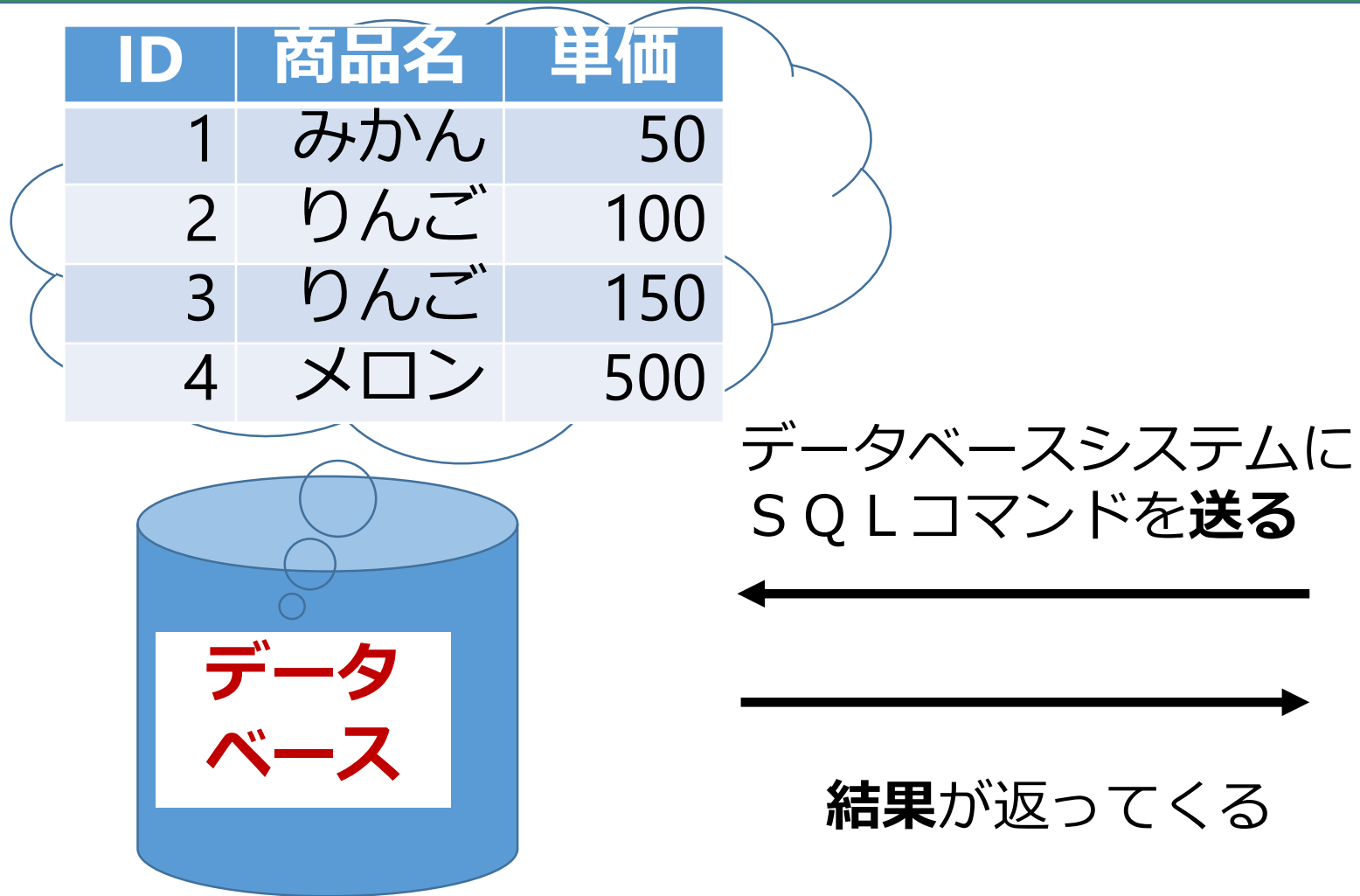
A(name, teacher_name), B(name, student_name, score)

name	teacher_name
Database	K
Programming	A

name	student_name	score
Database	KK	85
Database	AA	75
Database	LL	90
Programming	KK	85
Programming	LL	75



SQLの使い方



データベース
利用者

SQLで、テーブルをそのまま表示

ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500

元のテーブルの
まま表示

1	<code>select * from 商品;</code>
2	
3	
4	

ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500



問い合わせ
(クエリ)



結果

フィールドの表示／非表示（射影）の例

ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500

フィールドの表示／非表示 （「射影」といいます）



問い合わせ
(クエリ)

クエリ1
SELECT 商品名, 単価 FROM 商品;



商品名	単価
みかん	50
りんご	100
りんご	150
メロン	500

結果

レコードの絞り込み（選択）の例

ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500

全体で「射影＋選択」

レコードの絞り込み
(選択)



```
1 select 商品名, 単価 from 商品
2 where 単価 > 80;
3
```

商品名	単価
りんご	100
りんご	150
メロン	500

問い合わせ
(クエリ)

結果

※ 途中で改行してもいいし、改行しなくてもよい (SQL のルール)

結合の例

ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500

ID	名前	商品番号
1	X	3
2	Y	1

商品.ID	商品名	単価	購入.ID	名前	商品番号
1	みかん	50	1	X	3
1	みかん	50	2	Y	1
2	りんご	100	1	X	3
2	りんご	100	2	Y	1
3	りんご	150	1	X	3
3	りんご	150	2	Y	1
4	メロン	500	1	X	3
4	メロン	500	2	Y	1

```
select * from 商品, 購入;
```



問い合わせ
(クエリ)



結果

分解の例

講義名	担当教員	受講者
DB	K	CC
DB	K	AA
DB	K	BB
プロ	A	AA
プロ	A	DD

講義名	担当教員
DB	K
プロ	A

講義名	受講者
DB	CC
DB	AA
DB	BB
プロ	AA
プロ	DD

```
select DISTINCT 講義名 担当教員 from 授業;  
select DISTINCT 講義名 受講者 from 授業;
```

結果が2つ欲しい
とき、SQLも2つ



問い合わせ
(クエリ)



結果

SQL による問い合わせ（クエリ）

■ SQL による問い合わせ（クエリ）の結果は、
必ず 1 つのテーブルになる

	ID	商品名	単価
1	1	みかん	50
2	2	りんご	100
3	3	りんご	150
4	4	りんご	500

そのまま表示

	商品名	単価
1	りんご	100
2	りんご	150
3	りんご	500

射影＋選択

	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	りんご	500

射影

1	みかん	50	1	X	3
1	みかん	50	2	Y	1
2	りんご	100	1	X	3
2	りんご	100	2	Y	1
3	りんご	150	1	X	3
3	りんご	150	2	Y	1
4	りんご	500	1	X	3
4	りんご	500	2	Y	1

結合

SQL を用いた問い合わせ（クエリ）

- 「**問い合わせ（クエリ）**」とは、データベースの**検索（抽出）**、**分類**、**集計・集約**を行うこと
- **問い合わせ（クエリ）**の結果は、**必ず1つ**のテーブルになる
- キーワード: **select, from, where**

SQL 問い合わせの例

select * from 商品;

そのまま表示

select 商品名, 単価 from 商品;

射影

select 商品名, 単価 from 商品 where 単価 > 80;

射影 + 選択

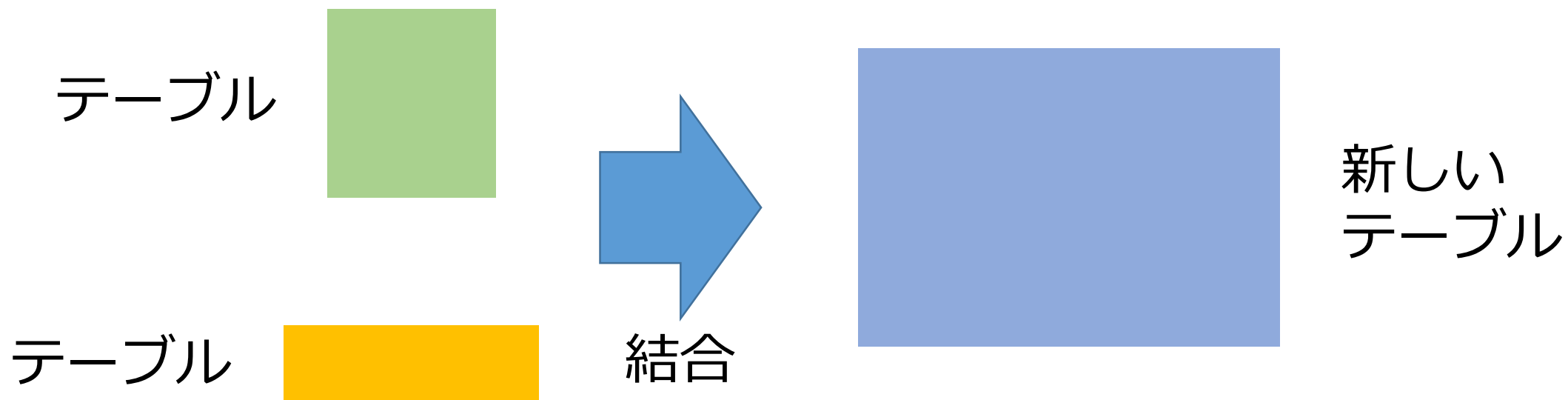
select * from 商品, 購入;

結合

6-2 結合

結合

2つのテーブルを所定の条件（結合条件という）
で、1つにまとめる。これが結合



複数のテーブルを1つにまとめる（結合）例

商品

ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500

購入

ID	名前	商品
1	X	3
2	Y	1

SQL でテーブル名を2つ書く

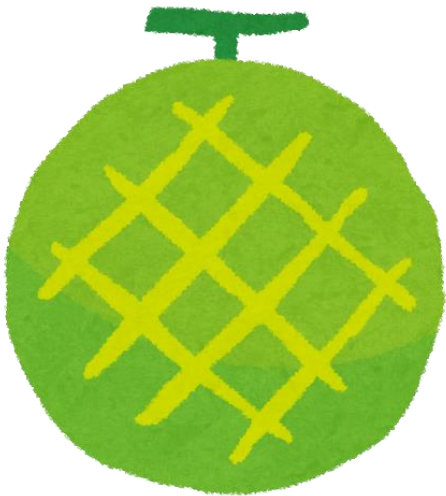
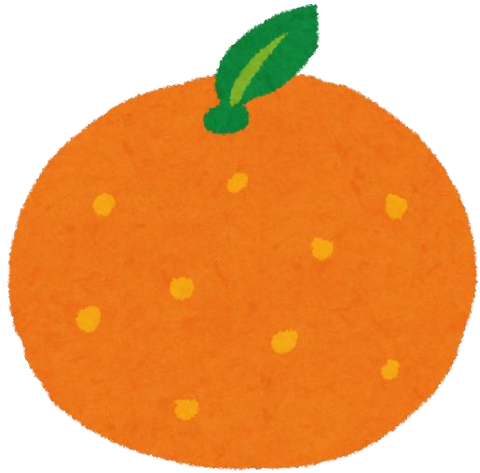
```
select * from 商品, 購入;
```



1	みかん	50	1	X	3
1	みかん	50	2	Y	1
2	りんご	100	1	X	3
2	りんご	100	2	Y	1
3	りんご	150	1	X	3
3	りんご	150	2	Y	1
4	りんご	500	1	X	3
4	りんご	500	2	Y	1

リレーショナルデータベースの結合結果では、
右側に一方のテーブルのレコードが、
左側に、もう一方のテーブルのレコードが集まる

商品



1つのテーブルには、同じ種類のデータが集まっている

商品

ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500

購入



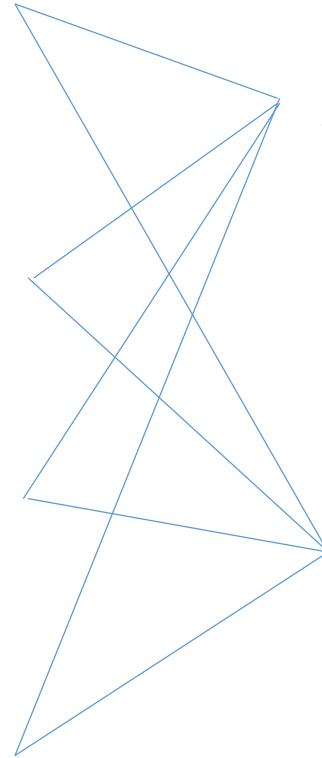
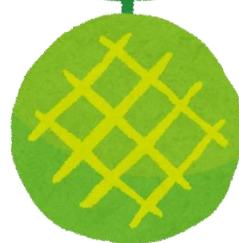
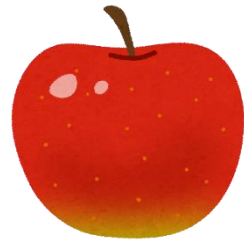
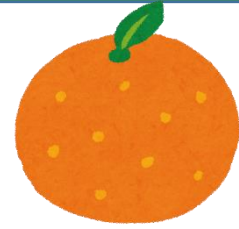
1つのテーブルには、同じ種類のデータが集まっている

購入



ID	名前	商品番号
1	X	3
2	Y	1

2つのテーブルの結合



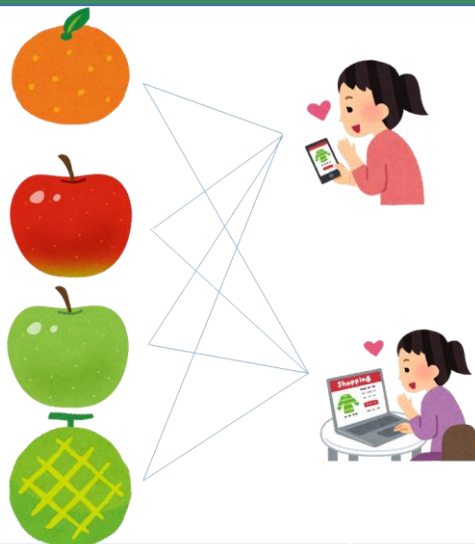
ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500

ID	名前	商品番号
1	X	3
2	Y	1

ペアは8通り

2つのテーブルの結合

ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500



ID	名前	商品番号
1	X	3
2	Y	1

商品.ID	商品名	単価	購入.ID	名前	商品番号
1	みかん	50	1	X	3
1	みかん	50	2	Y	1
2	りんご	100	1	X	3
2	りんご	100	2	Y	1
3	りんご	150	1	X	3
3	りんご	150	2	Y	1
4	メロン	500	1	X	3
4	メロン	500	2	Y	1

ペアは
8通り

「商品.ID」, 「購入.ID」って, 何でしょう?

「商品.」, 「購入.」 . . . テーブル名

結合するテーブルに「同じフィールド名」がある（バッティング）と、頭に、**自動**で、「テーブル名.」が付く。区別のため

商品.ID	商品名	単価	購入.ID	名前	商品番号
1	みかん	50	1	X	3
1	みかん	50	2	Y	1
2	りんご	100	1	X	3
2	りんご	100	2	Y	1
3	りんご	150	1	X	3
3	りんご	150	2	Y	1
4	メロン	500	1	X	3
4	メロン	500	2	Y	1

2つのテーブルの結合のための SQL コマンド

商品

```
select * from 商品, 購入;
```

ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500

購入

ID	名前	商品番号
1	X	3
2	Y	1



商品.ID	商品名	単価	購入.ID	名前	商品番号
1	みかん	50	1	X	3
1	みかん	50	2	Y	1
2	りんご	100	1	X	3
2	りんご	100	2	Y	1
3	りんご	150	1	X	3
3	りんご	150	2	Y	1
4	メロン	500	1	X	3
4	メロン	500	2	Y	1

6-3 結合条件

結合条件

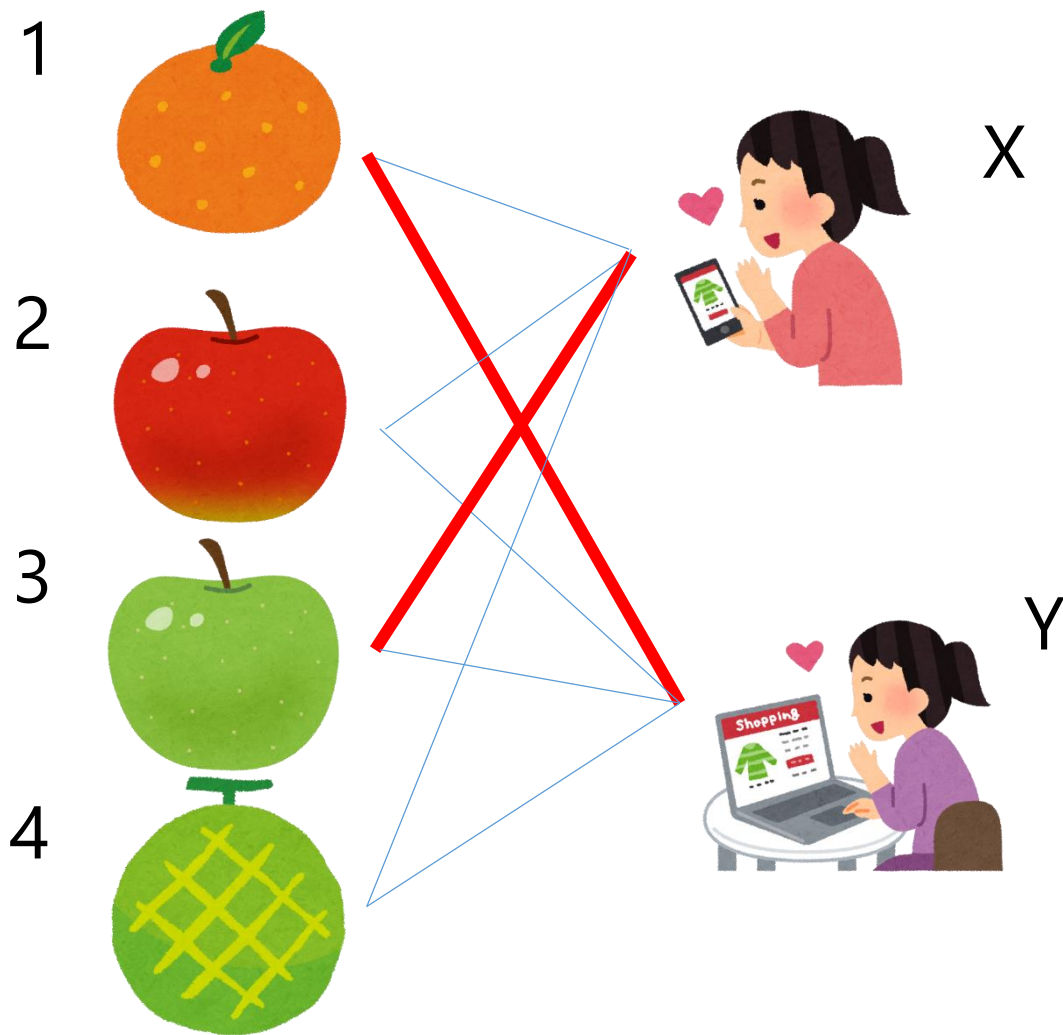
- ◆ 結合とは、2つのテーブルを結合して、1つの新しいテーブルを作るもの

例： `select * from 商品, 購入;`

- ◆ 結合では、結合条件を付けることができる

例： `select * from 商品, 購入
where 商品.ID = 商品番号;`

こんな場合を考えてみます



実際には、
Xさんは、**3**を買った
Yさんは、**1**を買った

こんな場合を考えてみます

商品

ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500

購入

ID	名前	商品番号
1	X	3
2	Y	1

実際には、

Xさんは、3 を買った

Yさんは、1 を買った

購入テーブルの情報

商品テーブルの情報

こんな場合を考えてみます

ID **商品**

ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500

購入

ID	名前	商品番号
1	X	3
2	Y	1

結合前

商品番号

結合条件付きの結合

```
select * from 商品, 購入
where 商品.ID = 商品番号;
```

商品.ID	商品名	単価	購入.ID	名前	商品番号
1	みかん	50	2	Y	1
3	りんご	150	1	X	3

実際にはどうだったかを「選択」している

結合だけ v s 結合 + 選択

```
select * from 商品, 購入;
```

商品.ID	商品名	単価	購入.ID	名前	商品番号
1	みかん	50	1	X	3
1	みかん	50	2	Y	1
2	りんご	100	1	X	3
2	りんご	100	2	Y	1
3	りんご	150	1	X	3
3	りんご	150	2	Y	1
4	メロン	500	1	X	3
4	メロン	500	2	Y	1

結合条件なし

```
select * from 商品, 購入  
where 商品.ID = 商品番号;
```

商品.ID	商品名	単価	購入.ID	名前	商品番号
1	みかん	50	2	Y	1
3	りんご	150	1	X	3

結合条件付き

結合だけ v s 結合 + 選択

```
select * from 商品, 購入;
```

商品.ID	商品名	単価	購入.ID	名前	商品番号
1	みかん	50	1	X	3
1	みかん	50	2	Y	1
2	りんご	100	1	X	3
2	りんご	100	2	Y	1
3	りんご	150	1	X	3
3	りんご	150	2	Y	1
4	メロン	500	1	X	3
4	メロン	500	2	Y	1

結合条件なし

```
select * from 商品, 購入  
where 商品.ID = 商品番号;
```

商品.ID	商品名	単価	購入.ID	名前	商品番号
1	みかん	50	2	Y	1
3	りんご	150	1	X	3

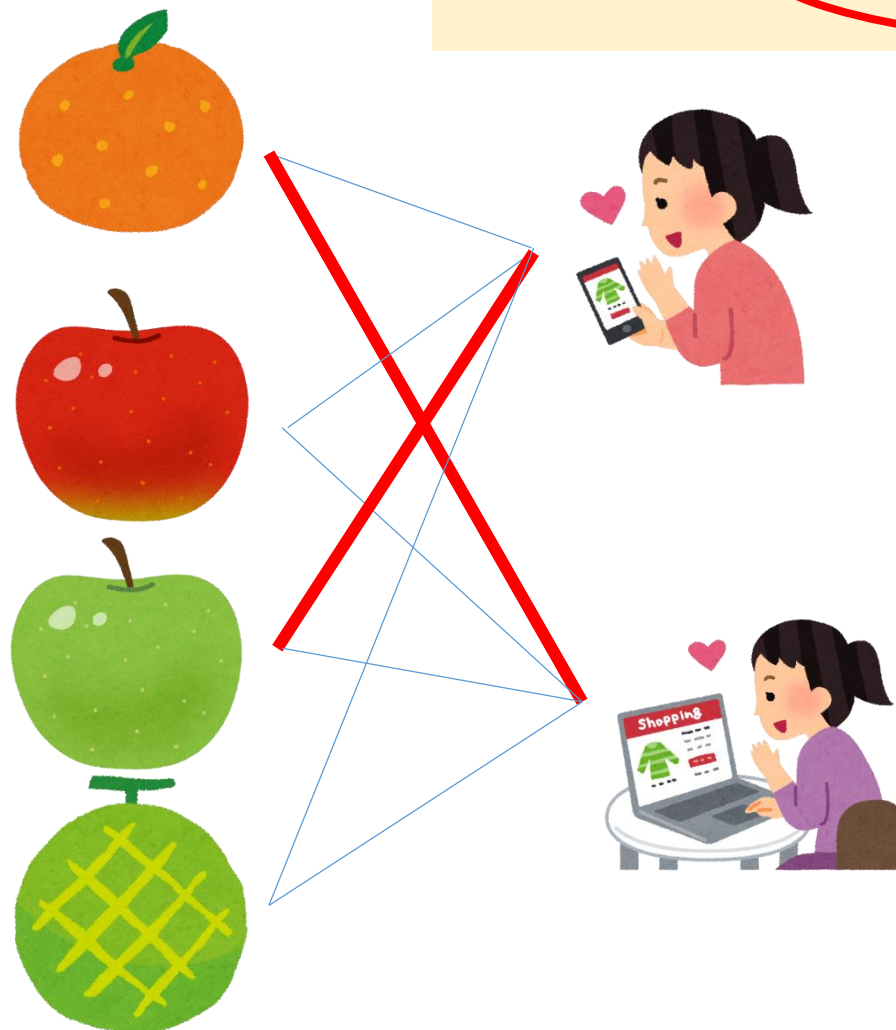
結合してから選択

結合条件付き

結合条件

```
select * from 商品, 購入  
where 商品.ID = 商品番号;
```

ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500



ID	名前	商品番号
1	X	3
2	Y	1

結合とは



ノートページ

結合とは

1. 2つのテーブルのレコードのすべてのペアを作る
2. 結合条件が指定されているときは、「1」の中から結合条件を満足するものだけを選ぶ

結合条件の例

結合条件：商品.ID = 商品番号

結合条件の書き方

＜テーブル名＞.＜フィールド名＞ = ＜テーブル名＞.＜フィールド名＞

※ 他のテーブルに同じフィールド名が無いときは、「テーブル名」を省略可

※ 「=」のところは： =, <>, <, <=, >, >= が可能

結合条件の書き方



ノートページ

結合条件の書き方

＜テーブル名＞.＜フィールド名＞ **=** ＜テーブル名＞.＜フィールド名＞

※ 他のテーブルに同じフィールド名が無いときは、「テーブル名」を省略可

※ 「**=**」のところは：**=**, **<>**, **<**, **<=**, **>**, **>=** が可能

6-4 SQL で結合を行う

結合を行う SQL の例

```
select * from 昼食, 価格 where 昼食.注文 = 価格.品名;
```

* または
フィールド名リスト

テーブル名リスト

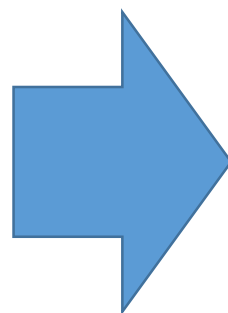
選択条件は where のあと

昼食

氏名	注文
XX	カレー
YY	うどん
ZZ	カレー

価格

品名	価格
カレー	500
うどん	400



氏名	注文	品名	価格
XX	カレー	カレー	500
YY	うどん	うどん	400
ZZ	カレー	カレー	500

結合の例

昼食

氏名	注文
XX	カレー
YY	うどん
ZZ	カレー

価格

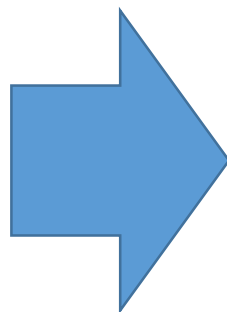
品名	価格
カレー	500
うどん	400

結合前

```
select * from 昼食, 価格  
where 昼食.注文 = 価格.品名;
```



ノートページ



結合

氏名	注文	品名	価格
XX	カレー	カレー	500
YY	うどん	うどん	400
ZZ	カレー	カレー	500

結合条件付き の結合

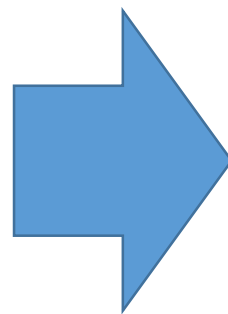
こんな書き方もできます

ID

ID	商品名	単価
1	みかん	50
2	りんご	100
3	りんご	150
4	メロン	500

ID	名前	商品番号
1	X	3
2	Y	1

```
select 商品名, 名前 from 商品, 購入  
where 商品.ID = 商品番号;
```



商品名	名前
みかん	Y
りんご	X

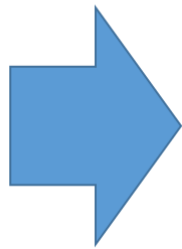
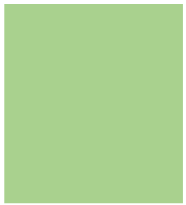
結果

リレーショナルデータベースでは

問い合わせ（クエリ）のときに、
テーブルは結合できる

そう思って設計すると、
良いデータベース設計ができる！

テーブル



結合



新しい
テーブル

