6. テーブルの結合

JOIN、結合と SQL 問い合わせ

URL: https://www.kkaneko.jp/de/ds/index.html

金子邦彦

謝辞：この資料では「いらすとや」のイラストを使用しています

① 論理的思考と問題解決能力

② データ分析と洞察の獲得

③ データ管理とSQLスキルの強化

アウトライン

1. イントロダクション
2. 結合の基本概念
3. SQL での結合の書き方
4. 結合条件のない結合と、結合条件のある結合
5. 演習

SQLFiddle のサイトにアクセス

Webブラウザを使用

1. ウェブブラウザを開く
2. アドレスバーにSQLFiddleのURLを入力

http://sqlfiddle.com/

URLが分からないときは、Googleなどの**検索エンジン**を利用。
「**SQLFiddle**」と**検索**し、表示された結果からSQLFiddleの
ウェブサイトをクリック。

SQLFiddle の画面

- **左側のパネル: テーブル定義、データの追加**など。SQLのCREATE TABLE や INSERT INTO などを入力。

- **右側のパネル: SQL問い合わせ。** SELECT などを入力。

SQL Fiddle MySQL 5.6

```
CREATE TABLE employees (
    id INTEGER,
    name TEXT,
    age INTEGER,
    salary INTEGER,
    department_id INT);
INSERT INTO employees VALUES (1, 'Alice', 30, 50000, 1);
INSERT INTO employees VALUES (2, 'Bob', 40, 60000, 1);
INSERT INTO employees VALUES (3, 'Charlie', 35, 70000, 2);
```

Build Schema | Edit Fullscreen | Browser | [;]

```
SELECT * FROM employees;
```

Run SQL | Edit Fullscreen | [;]

- **実行ボタン**

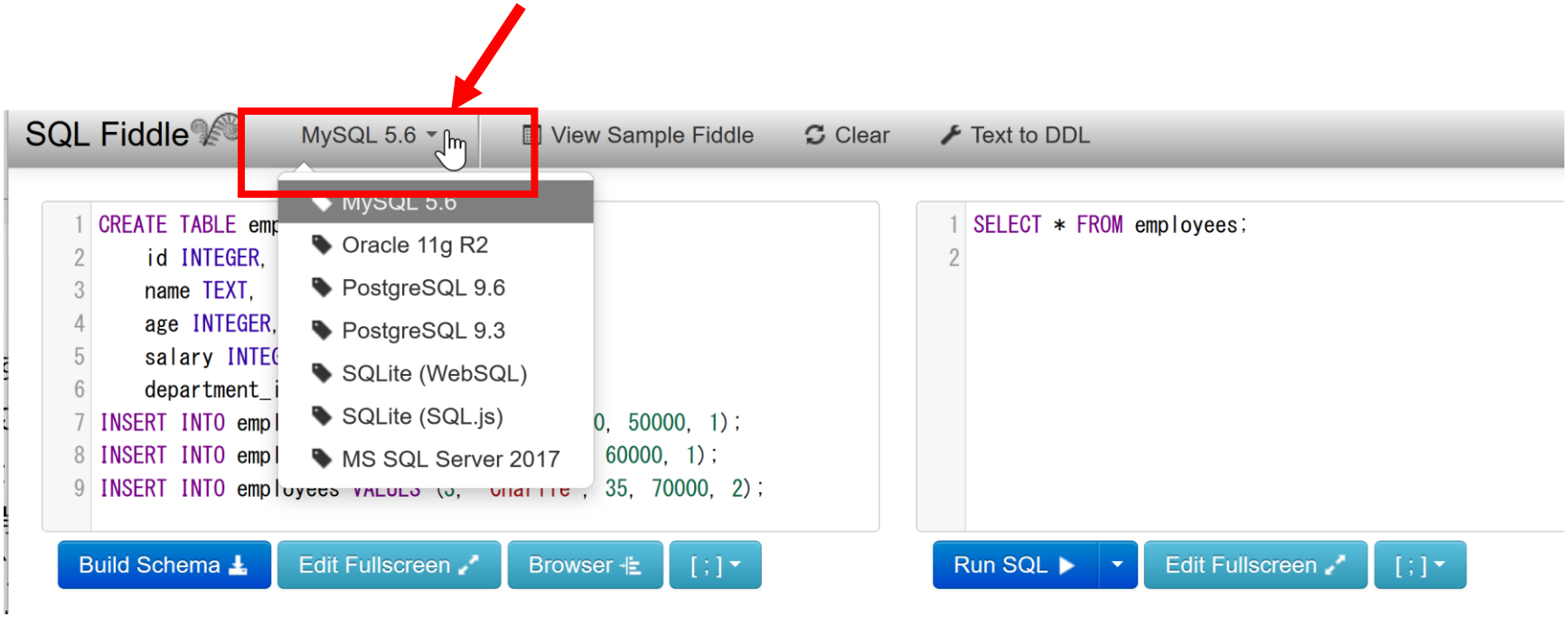
- **実行ボタン**

id	name	age	salary	department_id
1	Alice	30	50000	1
2	Bob	40	60000	1
3	Charlie	35	70000	2

結果ウィンドウ

SQLFiddle でのデータベース管理システムの選択（高度な機能）

データベース管理システムの選択
（この授業では MySQL を使用）



6-1. イントロダクション

リレーショナルデータベースの仕組み

- データを**テーブル**と呼ばれる**表形式で保存**
- **テーブル間**は**関連**で結ばれる。複雑な構造を持ったデータを効率的に管理することを可能に。

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

関連

ID	購入者	商品ID	数量
1	X	1	10
2	Y	2	5

リレーショナルデータベースの重要性

1. **データの整合性**: リレーショナルデータベースは、**データの整合性を保持するための機能**を有する。これにより、誤ったデータや矛盾したデータが保存されるのを防ぐことができる。
2. **柔軟な問い合わせ（クエリ）能力**: リレーショナルデータベースの**SQL**（Structured Query Language）の使用により、**複雑な検索やデータの抽出**が可能になる。
3. **トランザクション**の機能: 一連の操作全体を一つの単位として取り扱うことができる機能。これにより、**データの一貫性と信頼性が向上**する。
4. **セキュリティ**: **アクセス権限の設定**などにより、セキュリティを確保。

データの安全な保管、効率的なデータ検索・操作、ビジネスや研究の意思決定をサポート。

SQL 理解のための前提知識

○ テーブル

データを**テーブル**と呼ばれる**表形式で保存**

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

購入者	商品番号
X	1
X	3
Y	2

○ 問い合わせ（クエリ）

- **問い合わせ（クエリ）**は、**データベース**から**必要なデータを検索、加工するための指令**
- SELECT, FROM, WHERE など、**多様**なコマンドが存在。
- **結合、集計、ソート、副問い合わせ**など、高度な操作も可能

テーブル「商品」からデータを取得するSQLの例

- **SELECT** * **FROM** 商品;
- **SELECT** 商品名 **FROM** 商品;
- **SELECT DISTINCT** 商品名 **FROM** 商品;
- **SELECT** 商品名, 単価 **FROM** 商品 **WHERE** 単価 > 80;
- **SELECT** 商品名, 単価 **FROM** 商品 **WHERE** 単価 **BETWEEN** 80 **AND** 85;
- **SELECT AVG**(単価) **FROM** 商品;

  AVG, MAX, MIN, SUM: 平均、最大、最小、合計

  COUNT: 行数
- **SELECT** * **FROM** 商品 **WHERE** 商品名 **LIKE** '%ん';

  「*ん」、「ん*」「*ん*」のように書くことができる

  Access では % でなく *
- **SELECT** * **FROM** 商品 **WHERE** 商品名 **IN** ('みかん', 'りんご');

SQL によるテーブル定義

- **テーブル名**：**商品**
- **属性名**：**ID、商品名、単価**
- **属性のデータ型**：**数値、テキスト、数値**
- データの整合性を保つための**制約**：**なし**

```
CREATE TABLE 商品 (
    ID INTEGER,
    商品名 TEXT,
    単価 INTEGER);
```

データ追加のSQL

商品

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

```
INSERT INTO 商品 VALUES(1, 'みかん', 50);
INSERT INTO 商品 VALUES(2, 'りんご', 100);
INSERT INTO 商品 VALUES(3, 'メロン', 500);
```

演習１．テーブル定義とデータの追加

【トピックス】

1. SQL によるテーブル定義
2. SQL によるデータの追加
3. 問い合わせ（クエリ）による確認

Webブラウザを使用

① アドレスバーにSQLFiddleのURLを入力

http://sqlfiddle.com/

URLが分からないときは、Googleなどの**検索エンジン**を利用。「**SQLFiddle**」と**検索**し、表示された結果からSQLFiddleのウェブサイトをクリック。

② **左側のパネル**に、**テーブル定義**と**データの追加**を行う SQL を入れる。

```
CREATE TABLE 商品 (
    ID INTEGER,
    商品名 TEXT,
    単価 INTEGER);
INSERT INTO 商品 VALUES(1, 'みかん', 50);
INSERT INTO 商品 VALUES(2, 'りんご', 100);
INSERT INTO 商品 VALUES(3, 'メロン', 500);
```

③ 「Build Schema」をクリック

SQL Fiddle　MySQL 5.6　View Sample Fiddle　Clear　Text to DDL

```
CREATE TABLE 商品 (
    ID INTEGER,
    商品名 TEXT,
    単価 INTEGER);
INSERT INTO 商品 VALUES(1, 'みかん', 50);
INSERT INTO 商品 VALUES(2, 'りんご', 100);
INSERT INTO 商品 VALUES(3, 'メロン', 500);
```

Build Schema　Edit Fullscreen　Browser　[;]

Please build schema.

Run SQL　Edit Fullscreen　[;]

Please build schema.

④ **右側のパネル**に、**問い合わせ（クエリ）**を行う SQL を入れる。

```
select * FROM 商品;
```

⑤ 「**Run SQL**」をクリック

SQL 文が**実行**され、結果が表示される。

⑥ 下側のウインドウで、**結果を確認。**

6-2. 結合の基本概念

結合とその目的

結合は、**異なるテーブルを結合**して、**新たなテーブルを生成**する操作

- **主な目的**： データベース内の異なるテーブルから**データを組み合わせ**て、**有用なデータを作成**
- **結合条件**：**結合条件**は**通常**、**2つのテーブルの特定の属性同士の値が等しい**という条件を指定。**その他の複雑な条件**も指定できる

結合を行うことで、データベースの柔軟性と効率性を向上。

意思決定や問題解決に役立つデータを得ることができる。

商品テーブルと購入テーブル

商品

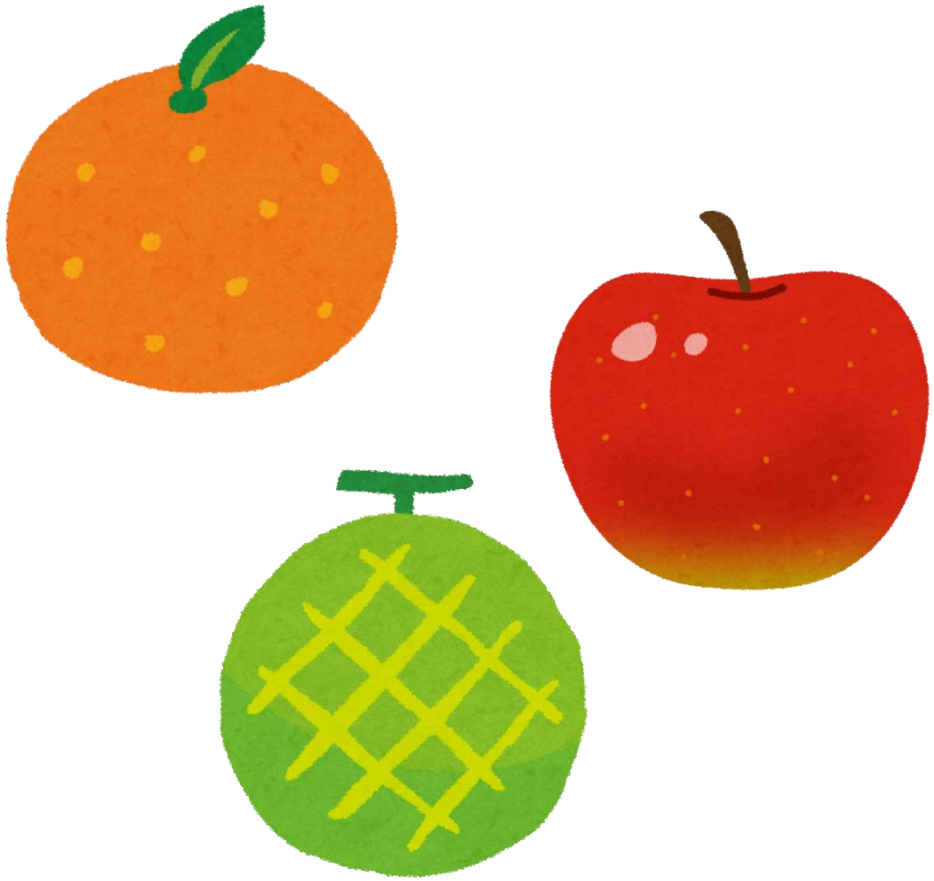
ID	商品名	単価
1	みかん	50
2	りんご	100
3	メロン	500

関連

購入

購入者	商品番号
X	1
X	3
Y	2

商品



商品

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

購入

購入

購入者	商品番号
X	1
X	3
Y	2

商品テーブルと購入テーブル

商品

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

関連

購入

購入者	商品番号
X	1
X	3
Y	2

Xさんは、**1** の**みかん**と,
3 の**メロン**を買った
Yさんは、 **2** の**りんご**を買った

購入テーブルの情報

商品テーブルの情報

結合と結合条件

商品

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

関連

購入

購入者	商品番号
X	1
X	3
Y	2

結合条件

「**商品テーブルのIDと購入テーブルの商品番号が等しい**」という結合条件を考えることができる

結合結果の例

結合結果は新しいテーブル

結合の有用性

- **どの購入者がどの商品を購入したかを調べる**など。
- さまざまなテーブルを組み合わせて、有用なデータを得て、分析できる。

結合の例

商品

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

関連

購入

購入者	商品番号
X	1
X	3
Y	2

結合のためのSQL

```
SELECT * FROM 商品
JOIN 購入
ON 商品.ID = 購入.商品番号;
```

結合条件

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

SQLの実行により
新しく生成される
テーブル

演習２．SQL による結合

【トピックス】

1. 結合
2. JOIN と ON の使用

Webブラウザを使用

① アドレスバーにSQLFiddleのURLを入力

http://sqlfiddle.com/

URLが分からないときは、Googleなどの**検索エンジン**を利用。「**SQLFiddle**」と**検索**し、表示された結果からSQLFiddleのウェブサイトをクリック。

② **左側のパネル**に、**テーブル定義**と**データの追加**を行う SQL を入れる。（以前の SQL は不要なので**消す**）

```
CREATE TABLE 商品 (
    ID INTEGER,
    商品名 TEXT,
    単価 INTEGER);
INSERT INTO 商品 VALUES(1, 'みかん', 50);
INSERT INTO 商品 VALUES(2, 'りんご', 100);
INSERT INTO 商品 VALUES(3, 'メロン', 500);
CREATE TABLE 購入 (
    購入者 TEXT,
    商品番号 INTEGER);
INSERT INTO 購入 VALUES('X', 1);
INSERT INTO 購入 VALUES('X', 3);
INSERT INTO 購入 VALUES('Y', 2);
```

③ 「Build Schema」をクリック

SQL Fiddle　MySQL 5.6　View Sample Fiddle　Clear　Text to DDL

```
CREATE TABLE 商品 (
    ID INTEGER,
    商品名 TEXT,
    単価 INTEGER);
INSERT INTO 商品 VALUES(1, 'みかん', 50);
INSERT INTO 商品 VALUES(2, 'りんご', 100);
INSERT INTO 商品 VALUES(3, 'メロン', 500);
CREATE TABLE 購入 (
    購入者 TEXT,
    商品番号 INTEGER);
INSERT INTO 購入 VALUES('X', 1);
INSERT INTO 購入 VALUES('X', 3);
INSERT INTO 購入 VALUES('Y', 2);
```

Please build schema.

Build Schema　Edit Fullscreen　Browser　[;]　Run SQL　Edit Fullscreen　[;]

Schema Ready

Please build schema.

④ **右側のパネル**に、**問い合わせ（クエリ）**を行う SQL を入れる。（以前の SQL は不要なので**消す**）

```
SELECT * FROM 商品
JOIN 購入
ON 商品.ID = 購入.商品番号;
```

⑤ 「**Run SQL**」をクリック

SQL 文が**実行**され、結果が表示される。

⑥ 下側のウインドウで、**結果を確認。**

演習２の実行結果

商品

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

関連

購入

購入者	商品番号
X	1
X	3
Y	2

- 商品テーブルと購入テーブルを**結合**して、**購入者がどの商品を購入したかのデータ**を取得。
- **結合条件**は、**商品テーブルのID属性と購入テーブルの商品番号属性が等しい**場合に結合

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

```
SELECT * FROM 商品
JOIN 購入
ON 商品.ID = 購入.商品番号;
```

結合の利点

結合はリレーショナルデータベースにおいて重要な操作であり、以下の利点がある

- **データの統合**：異なるテーブルを結合して、新たなテーブルを生成。これにより、関連のあるデータを１つにまとめるデータ統合を実現。
- **データの洞察**：異なるデータを組み合わせて意思決定に役立つ洞察を得る。
- **データの整合性：**データの冗長性を排除し、データの整合性を保つために、テーブルを分割して、データベース内に保持。必要に応じて結合して必要なデータを取得できる。

ここまでのまとめ

結合は、**異なるテーブルを結合**して、**新たなテーブルを生成**する操作

- **主な目的**： データベース内の異なるテーブルから**データを組み合わせ**て、**有用なデータを作成**
- **結合条件**は、例えば、「**商品テーブルのIDと購入テーブルの商品番号が等しい**」というような結合条件を考えることができる
- **有用性**：結合を利用することで、**購入者がどの商品を購入したか**などのデータを得ることができる
- 結合を行う SQL の例

```
SELECT * FROM 商品
JOIN 購入
ON 商品.ID = 購入.商品番号;
```

自習 1

- **目的:** 結合の結果をさらに加工する方法をマスターする
- **指示:** 演習1の結果のテーブルは5列です。このうち、**「商品名」と「購入者」の列のみを表示し、他の列は表示しないような（下図のように） SQL を作成してください。**

商品名	購入者
みかん	X
メロン	X
りんご	Y

- **ヒント:** SELECT * を変更して、必要な列のみを指定してください。

自習 2

- **目的:** 結合の結果をさらに加工する方法をマスターする
- **指示:** 演習1の結果のテーブルは、3行のテーブルです。**この行数3を得る（下図のように）SQL を作成してください**

COUNT(*)
3

- **ヒント:** COUNT(*) を使用してください。

自習１の解答例

```
SELECT 商品名, 購入者 FROM 商品
JOIN 購入
ON 商品.ID = 購入.商品番号;
```

自習２の解答例

```
SELECT COUNT(*) FROM 商品
JOIN 購入
ON 商品.ID = 購入.商品番号;
```

6-3. SQL での結合の書き方

結合のための JOIN と ON

```
SELECT * FROM 商品
JOIN 購入
ON 商品.ID = 購入.商品番号;
```

JOIN

- JOIN は、**テーブルの結合を実行するため**に使用
- 異なるテーブルの結合を可能にする

ON

- ON は、 JOINキーワードと一緒に使用される
- **結合条件を指定**に使用
- 結合条件は、通常、結合したいテーブルの特定の属性同士の値が等しい場合に使用される。複雑な条件も指定可能

結合条件の指定

結合のためのSQL

```
SELECT * FROM 商品
JOIN 購入
ON 商品.ID = 購入.商品番号;
```

} **結合条件**

- **商品テーブルの「ID」と購入テーブルの「商品番号」属性が等しい**という結合条件

商品.ID = 購入.商品番号

- **「等しい値を持つ」という結合条件の表し方**

テーブル1.属性3 = テーブル2.属性4

複数の条件の指定

- **複数の条件を組み合わせる**ために、**ANDやORを使用**

  AND　すべての条件が真（成り立つこと）が条件

  OR　いずれかの条件が真（成り立つこと）が条件

商品テーブルと購入テーブルを結合し、特定の購入者が特定の商品を購入した場合に結合する場合の例

商品.ID = 購入.商品番号 **AND** 購入.購入者 = 'X'

より詳細なデータの絞り込みが可能となり、必要なデータを正確に取得できるようになる。

演習３．複数の条件の指定

【トピックス】

1. 結合
2. 複数の結合条件
3. AND

Webブラウザを使用

① アドレスバーにSQLFiddleのURLを入力

http://sqlfiddle.com/

URLが分からないときは、Googleなどの**検索エンジン**を利用。「**SQLFiddle**」と**検索**し、表示された結果からSQLFiddleのウェブサイトをクリック。

② **左側のパネル**に、**テーブル定義**と**データの追加**を行う SQL を入れる。（**「演習２」と同じものである**）

```
CREATE TABLE 商品 (
    ID INTEGER,
    商品名 TEXT,
    単価 INTEGER);
INSERT INTO 商品 VALUES(1, 'みかん', 50);
INSERT INTO 商品 VALUES(2, 'りんご', 100);
INSERT INTO 商品 VALUES(3, 'メロン', 500);
CREATE TABLE 購入 (
    購入者 TEXT,
    商品番号 INTEGER);
INSERT INTO 購入 VALUES('X', 1);
INSERT INTO 購入 VALUES('X', 3);
INSERT INTO 購入 VALUES('Y', 2);
```

③ 「Build Schema」をクリック

SQL Fiddle　MySQL 5.6　View Sample Fiddle　Clear　Text to DDL

```
CREATE TABLE 商品 (
    ID INTEGER,
    商品名 TEXT,
    単価 INTEGER);
INSERT INTO 商品 VALUES(1, 'みかん', 50);
INSERT INTO 商品 VALUES(2, 'りんご', 100);
INSERT INTO 商品 VALUES(3, 'メロン', 500);
CREATE TABLE 購入 (
    購入者 TEXT,
    商品番号 INTEGER);
INSERT INTO 購入 VALUES('X', 1);
INSERT INTO 購入 VALUES('X', 3);
INSERT INTO 購入 VALUES('Y', 2);
```

Please build schema.

Build Schema　Edit Fullscreen　Browser　[;]　Run SQL　Edit Fullscreen　[;]

Schema Ready

Please build schema.

④ **右側のパネル**に、**問い合わせ（クエリ）**を行う SQL を入れる。（以前の SQL は不要なので**消す**）

```
SELECT * FROM 商品
JOIN 購入
ON 商品.ID = 購入.商品番号 AND 購入.購入者 = 'X';
```

⑤「**Run SQL**」をクリック

SQL 文が**実行**され、結果が表示される。

⑥ 下側のウインドウで、**結果を確認。**

⑦ **右側のパネル**に、**問い合わせ（クエリ）**を行う SQL を入れる。（以前の SQL は不要なので**消す**）

```
SELECT 商品名, 購入者, 単価 FROM 商品
JOIN 購入
ON 商品.ID = 購入.商品番号 AND 購入.購入者 = 'X';
```

⑧「**Run SQL**」をクリック

SQL 文が**実行**され、結果が表示される。

⑨ 下側のウインドウで、**結果を確認。**

自習 3

- **目的:** 複数の条件を指定するための AND をマスターする
- **指示:** 演習 3 において、次の SQL を実行したら、どのような結果になるか、予想してください。そして、実際に動作させてください

```
SELECT 商品名, 購入者, 単価 FROM 商品
JOIN 購入
ON 商品.ID = 購入.商品番号 AND 購入.購入者 = 'Y';
```

- **ヒント:**購入.購入者 = 'X' でなく、購入.購入者 = 'Y' になっていることに注意

- 自習３の解答例

商品名	購入者	単価
りんご	Y	100

6-4. 結合条件のない結合と結合条件のある結合

関連性とリレーショナルデータベースのテーブル

- **関連性**：異なるデータ間のつながりや対応関係を示す

  例：2つのデータセット **{1, 2, 3}** と **{a, b}**

  対応関係 **1-a, 2-b, 3-a**

- 関連性をリレーショナルデータベースのテーブルで扱うとき、**１つの対応が、テーブルの１行**になる

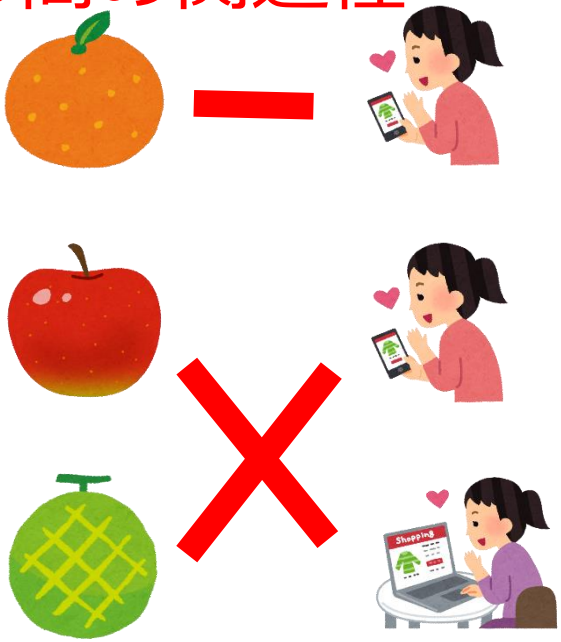
1	a
2	b
3	a

1つ目の列は {1, 2, 3} の要素。2つ目の列は {a, b} の要素。

- **テーブルを使用**することで、**関連性**を見やすく整理し、明確にできる。

２つのテーブルの間の関連性

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



	購入者	商品番号
ⓐ	X	1
ⓑ	X	3
ⓒ	Y	2

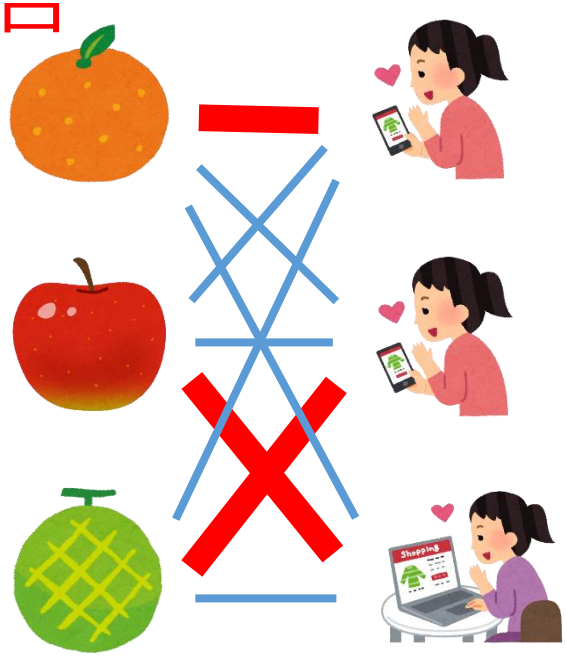
関連性　①-ⓐ　②-ⓒ　③-ⓑ

①，②，③とⓐⓑⓒの間の関連性は、リレーショナルデータベースでは、次のように示す

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

結合条件のある結合

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



購入者	商品番号
X	1
X	3
Y	2

結合のためのSQL（結合条件あり）

```
SELECT * FROM 商品
JOIN 購入
ON 商品.ID = 購入.商品番号;
```

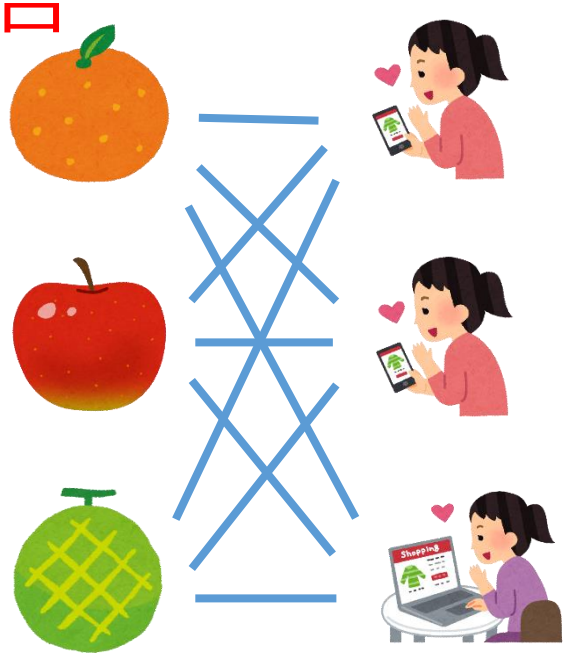
結合条件

結合の結果

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

結合条件のない結合

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



購入者	商品番号
X	1
X	3
Y	2

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

結合のためのSQL（結合条件なし）

```
SELECT * FROM 商品
JOIN 購入;
```

結合操作

- テーブルを結合することで、**新しいテーブルが生成**される

結合結果のテーブル

- 結合結果のテーブルは、**元の2つのテーブルの間の関係性**を示している
- データの結合によって、**関連性**が **1 つのテーブルの中**に、**明確に示される**ようになる。

結合条件の有無

- 結合条件が指定されない場合、元の2つのテーブルの間の**すべてのペアを含むテーブル**が作成される。

演習４．結合条件のない結合

【トピックス】

1. 結合
2. 結合条件のない結合

Webブラウザを使用

① アドレスバーにSQLFiddleのURLを入力

http://sqlfiddle.com/

URLが分からないときは、Googleなどの**検索エンジン**を利用。「**SQLFiddle**」と**検索**し、表示された結果からSQLFiddleのウェブサイトをクリック。

② **左側のパネル**に、**テーブル定義**と**データの追加**を行う SQL を入れる。（**「演習２」と同じものである**）

```
CREATE TABLE 商品 (
    ID INTEGER,
    商品名 TEXT,
    単価 INTEGER);
INSERT INTO 商品 VALUES(1, 'みかん', 50);
INSERT INTO 商品 VALUES(2, 'りんご', 100);
INSERT INTO 商品 VALUES(3, 'メロン', 500);
CREATE TABLE 購入 (
    購入者 TEXT,
    商品番号 INTEGER);
INSERT INTO 購入 VALUES('X', 1);
INSERT INTO 購入 VALUES('X', 3);
INSERT INTO 購入 VALUES('Y', 2);
```

③ 「Build Schema」をクリック

SQL Fiddle　MySQL 5.6　View Sample Fiddle　Clear　Text to DDL

```
CREATE TABLE 商品 (
    ID INTEGER,
    商品名 TEXT,
    単価 INTEGER);
INSERT INTO 商品 VALUES(1, 'みかん', 50);
INSERT INTO 商品 VALUES(2, 'りんご', 100);
INSERT INTO 商品 VALUES(3, 'メロン', 500);
CREATE TABLE 購入 (
    購入者 TEXT,
    商品番号 INTEGER);
INSERT INTO 購入 VALUES('X', 1);
INSERT INTO 購入 VALUES('X', 3);
INSERT INTO 購入 VALUES('Y', 2);
```

Please build schema.

Build Schema　Edit Fullscreen　Browser　[;]　Run SQL　Edit Fullscreen　[;]

Schema Ready

Please build schema.

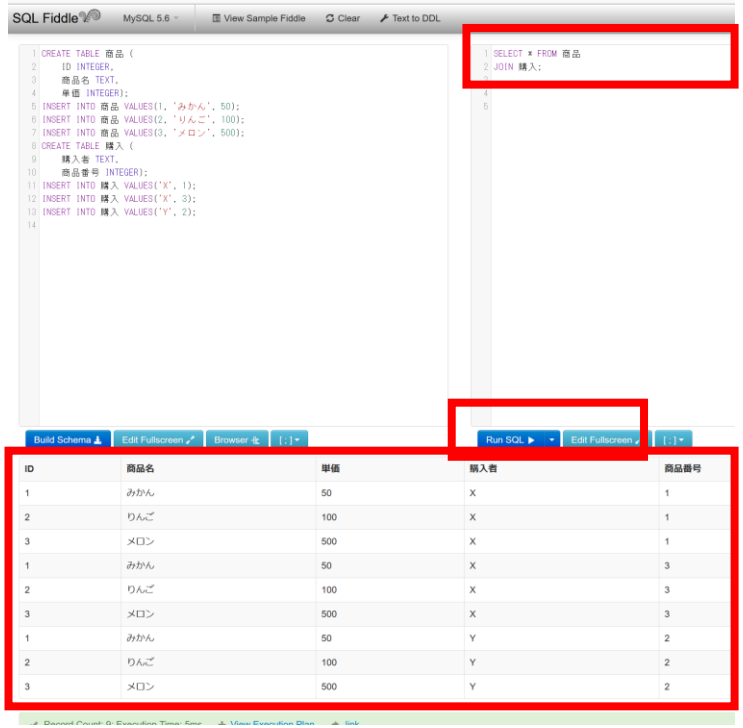
④ **右側のパネル**に、**問い合わせ（クエリ）**を行う SQL を入れる。（以前の SQL は不要なので**消す**）

```
SELECT * FROM 商品
JOIN 購入;
```

⑤「**Run SQL**」をクリック

SQL 文が**実行**され、結果が表示される。

⑥ 下側のウインドウで、**結果を確認。**



6-5. 演習

次の情報を扱う

- 織田　は　とうふ　を買った
- 豊臣　は　納豆　　を買った
- 徳川　は　納豆　　を買った

織田、豊臣、徳川に、通し番号の ID を付ける

　　１：織田、２：豊臣、３：徳川

とうふ、納豆に、通し番号の ID を付ける

　　１：とうふ、２：納豆

次の２つのテーブルを扱う

テーブル名：
名簿

ID	name	buy
1	織田	1
2	豊臣	2
3	徳川	2

関連

テーブル名：
食材

ID	name
1	とうふ
2	納豆

演習５．結合の演習

【トピックス】

1. 結合
2. 結合条件
3. JOIN、ON

Webブラウザを使用

① アドレスバーにSQLFiddleのURLを入力

http://sqlfiddle.com/

URLが分からないときは、Googleなどの**検索エンジン**を利用。「**SQLFiddle**」と**検索**し、表示された結果からSQLFiddleのウェブサイトをクリック。

② **左側のパネル**に、**テーブル定義**と**データの追加**を行う SQL を入れる。（以前の SQL は不要なので**消す**）

```
create table 名簿(
  ID integer,
  name text,
  buy integer
);
INSERT INTO 名簿 VALUES(1, '織田', 1);
INSERT INTO 名簿 VALUES(2, '豊臣', 2);
INSERT INTO 名簿 VALUES(3, '徳川', 2);
create table 食材(
  ID integer,
  name text
);
INSERT INTO 食材 VALUES(1, 'とうふ');
INSERT INTO 食材 VALUES(2, '納豆');
```

③ 「Build Schema」をクリック

SQL Fiddle　MySQL 5.6　View Sample Fiddle　Clear　Text to DDL

```
create table 名簿(
  ID integer,
  name text,
  buy integer
);
INSERT INTO 名簿 VALUES(1, '織田', 1);
INSERT INTO 名簿 VALUES(2, '豊臣', 2);
INSERT INTO 名簿 VALUES(3, '徳川', 2);
create table 食材(
  ID integer,
  name text
);
INSERT INTO 食材 VALUES(1, 'とうふ');
INSERT INTO 食材 VALUES(2, '納豆');
```

Build Schema　Edit Fullscreen　Browser　[;]

Please build schema.

Run SQL　Edit Fullscreen　[;]

Please build schema.

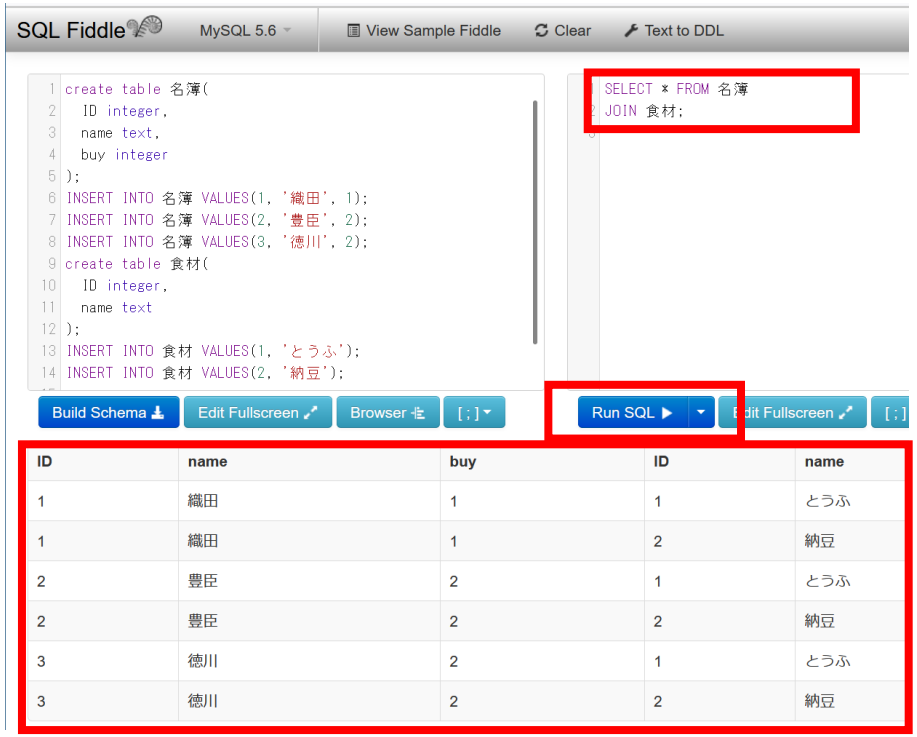
④ **右側のパネル**に、**問い合わせ（クエリ）**を行う SQL を入れる。（以前の SQL は不要なので**消す**）

```
SELECT * FROM 名簿
JOIN 食材;
```

⑤「**Run SQL**」をクリック

SQL 文が**実行**され、結果が表示される。

⑥ 下側のウインドウで、**結果を確認。**



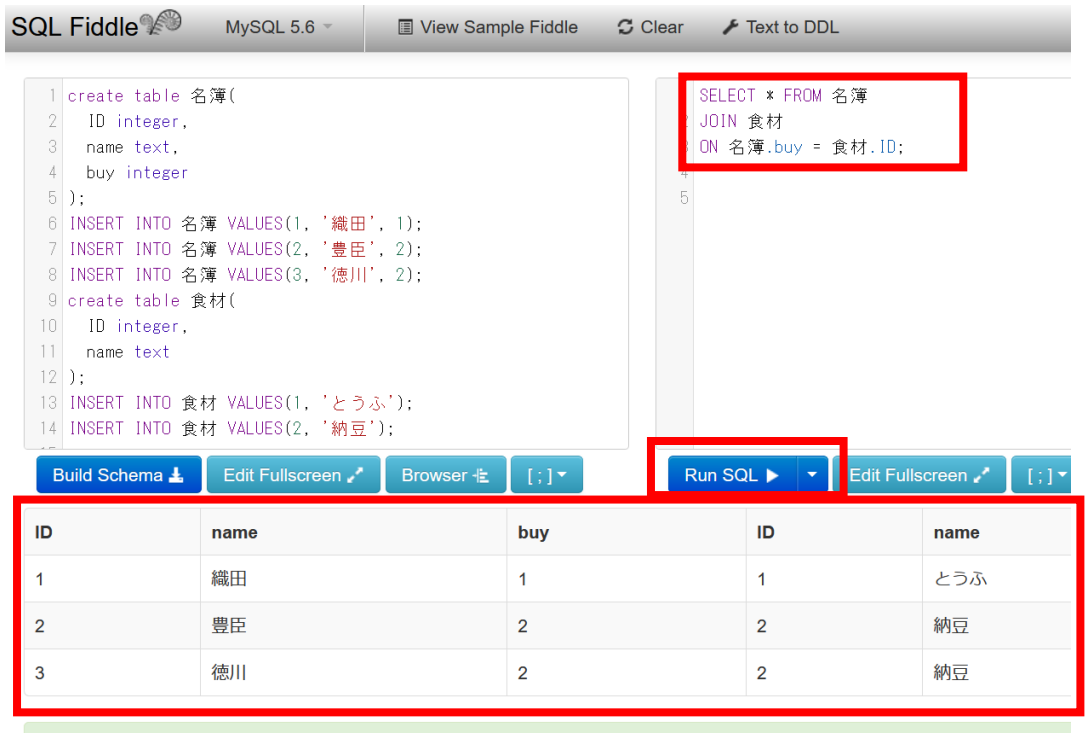
⑦ **右側のパネル**に、**問い合わせ（クエリ）**を行う SQL を入れる。（以前の SQL は不要なので**消す**）

```
SELECT * FROM 名簿
JOIN 食材
ON 名簿.buy = 食材.ID;
```

⑧「**Run SQL**」をクリック

SQL 文が**実行**され、結果が表示される。

⑨ 下側のウインドウで、**結果を確認。**



⑩ **右側のパネル**に、**問い合わせ（クエリ）**を行う SQL を入れる。（以前の SQL は不要なので**消す**）

```
SELECT 名簿.name, 食材.name FROM 名簿
JOIN 食材
ON 名簿.buy = 食材.ID;
```

⑪「**Run SQL**」をクリック

SQL 文が**実行**され、結果が表示される。

⑫ 下側のウインドウで、**結果を確認。**

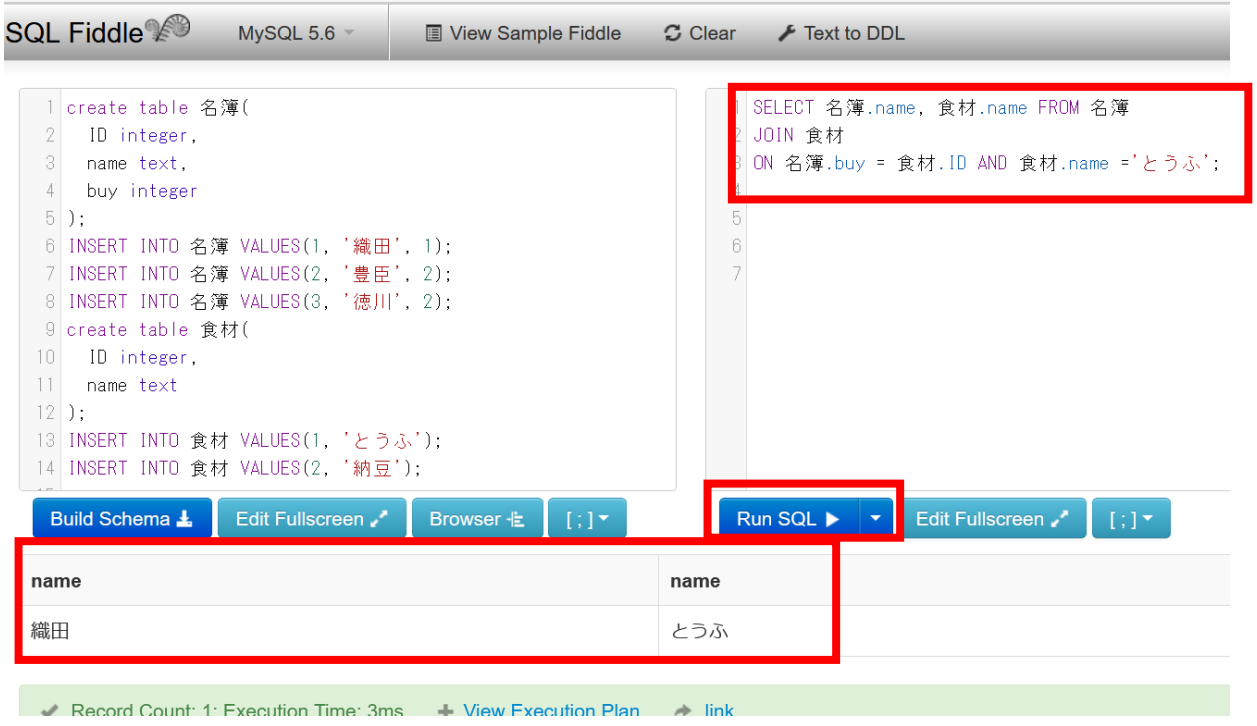
⑬ **右側のパネル**に、**問い合わせ（クエリ）**を行う SQL を入れる。（以前の SQL は不要なので**消す**）

```
SELECT 名簿.name, 食材.name FROM 名簿
JOIN 食材
ON 名簿.buy = 食材.ID AND 食材.name ='とうふ';
```

⑭「**Run SQL**」をクリック

SQL 文が**実行**され、結果が表示される。

⑮ 下側のウインドウで、**結果を確認。**



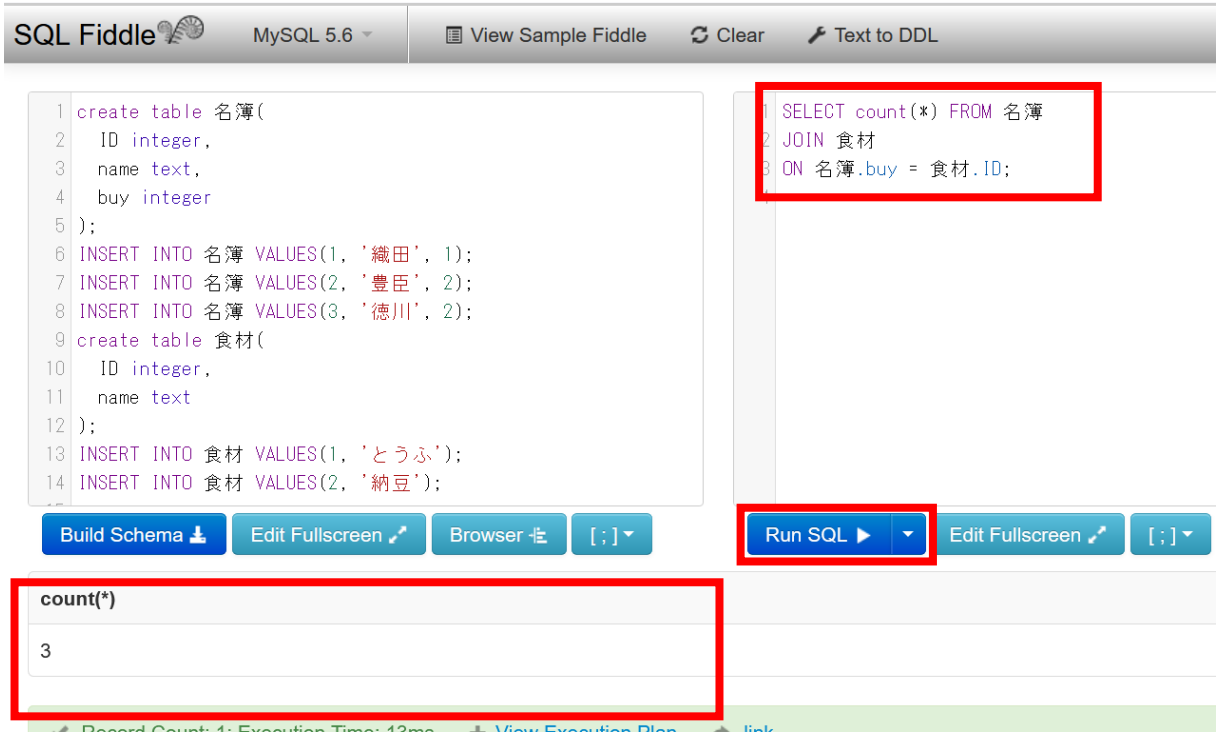
⑯ **右側のパネル**に、**問い合わせ（クエリ）**を行う SQL を入れる。（以前の SQL は不要なので**消す**）

```
SELECT count(*) FROM 名簿
JOIN 食材
ON 名簿.buy = 食材.ID;
```

⑰「**Run SQL**」をクリック

SQL 文が**実行**され、結果が表示される。

⑱ 下側のウインドウで、**結果を確認。**



全体まとめ

テーブルの結合の基本

- 結合は、異なるテーブルを一つにまとめる操作
- 結合条件は通常、テーブルの特定の属性同士の値が等しい場合に使用。その他、より複雑な結合条件などを指定できる。
- 結合によって、データベースの柔軟性と効率性が向上

SQLでの結合の書き方

- JOINとONを使用してテーブルの結合を行う
- SELECT * FROM 商品 JOIN 購入 ON 商品.ID = 購入.商品番号; のように書く
- 複数の条件を指定する場合は、ANDやORを使用して結合条件を追加

結合の応用

- 結合操作を通じて、関連あるデータを１つのテーブルに明確に示すことができるようになり、データの絞り込みや分析が容易になる。
- 結合条件を工夫することで、さまざまなビジネス要件に対応したデータの抽出や分析が可能。

① 論理的思考と問題解決能力

テーブルの結合を学ぶことで、異なるテーブルからから必要な情報を組み合わせ、欲しいデータを得て、具体的な問題を解決する能力が身につきます。この過程で、論理的思考の能力が向上します。

② データ分析と洞察の獲得

複数のテーブルから必要な情報を得て、分析する技術を習得します。データを用いて、より深い洞察を得て、意思決定ができるようになります。

③ データ管理とSQLスキルの強化

結合をSQLで実行するスキルを学ぶことで、効率的にデータを管理・操作するスキルが向上します。