

# データベース

ウェブサービスで利用される会員情報や、購買情報といった「データ」は、データベースに格納され、PHPなどのプログラムがそのデータベースにアクセスして、情報の追加や更新、削除などを行っています。そのデータベースを管理するためのソフトウェアがMySQLです。

## ● データベース

- 特定のテーマに沿ったデータを集めて管理し、容易に検索・抽出などの再利用をできるようにしたもの。 ... *Wikipedia*
- **列と行**の二次元でデータを表現
  - ✓ Excelと同じ！

| tweet_id | account  | contents | input_datetime      |
|----------|----------|----------|---------------------|
| 1        | junchiba | テストです    | 2014-11-22 10:10:10 |
| 2        | jiro     | おはよう     | 2014-11-22 08:10:23 |

列 (カラム)

(レコード)

## ● SQL(エスキューエル)

- データの操作や定義を行うためのデータベース言語（問い合わせ言語）である。

... Wikipedia

- プログラム言語の中の1つで、  
ほぼ**全ての業務システムに利用**されている。

```
SELECT * FROM consumer_tbl WHERE consumer_id = 1
```

抽出する

どこから

どんな条件で

# システム全体の中の位置づけ

1 情報を入力する



**HTML**

2 HTMLから情報をもろう  
& SQLを実行する



**PHPなど**

3 情報を登録する



**SQL**

# SQLを実行してみよう

## ● やってみよう

- データベース管理のためのアプリケーションである  
**phpMyAdmin**を使って**SQLを実行**してみましょう

✓ <http://db.zeropuro-study.online/phpmyadmin>

✓ IDとパスワードはuserXXXです。(XXXはマイフォルダ)

```
SELECT * FROM consumer_tbl WHERE consumer_id = 1
```

抽出する

どこから

どんな条件で

| consumer_tbl |            | product_tbl | deal_tbl |
|--------------|------------|-------------|----------|
| consumer_id  | tel        | name        | price    |
| 1            | 0801292... | yamada      |          |
| 2            | 0909311... | suzuki      |          |

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

- データベースはあくまで、「データの集合体」ではないため、それを**管理するための仕組み**が必要。
- そこで使われるのが**DBMS**。
- 現在、DBMSとして主に以下が知られています。

MySQL



無料

Webアプリケーションで主流

Oracle

ORACLE®

有料（3～500万円以上）

基幹業務システムなどで圧倒的なシェア

Microsoft Access



有料（1.5～6万円前後）

Windows上で動作 / 業務アプリケーション開発等で利用

ORACLE®

Microsoft®

# テーブル構造

## 前提条件

- どの顧客が、どの商品を、買っているかを記憶する  
顧客管理データベースを事例として考える

| 顧客ID | email   |
|------|---------|
| 1    | tanaka@ |
| 2    | sato@   |

顧客  
テーブル

| 製品ID | 名前      |
|------|---------|
| 1    | バスタオル   |
| 2    | フェイスタオル |

製品  
テーブル

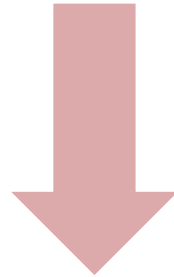
| 取引ID | 顧客ID | 製品ID | 個数 |
|------|------|------|----|
| 1    | 1    | 1    | 4  |
| 2    | 1    | 2    | 4  |
| 3    | 2    | 1    | 10 |

取引  
テーブル

## 問題その1.

すべての製品情報を表示したい。

どのようなプログラムを書けばよいか？



```
SELECT * FROM product_tbl
```



## ● SQL言語の書き方

抽出する

どこから

```
SELECT * FROM product_tbl
```

- 大文字と小文字の区別はありません。

## ● SQL言語の書き方 (条件付き)

```
SELECT * FROM product_tbl  
WHERE product_id = 1
```

どんな条件で

- 条件を指定するときはWhereをつける

## ● SQL言語の書き方(条件付き)

```
SELECT * FROM product_tbl  
WHERE product_name = 'バスタオル'
```

シングルクォーテーションが必要！

- 条件に文字列を使用する場合は、  
シングルクォーテーション（'）で囲む。

## ● 複数条件による検索(1/2)

```
SELECT * FROM product_tbl  
WHERE product_name = 'バスタオル'  
      AND product_id= 1
```

かつ

- **AND** は指定された  
全ての条件を満たすレコードが  
取得されます

## ● 複数条件による検索(2/2)

```
SELECT * FROM product_tbl  
WHERE product_name = 'バスタオル'  
      OR product_id= 1
```

または

- **OR**は指定された  
いずれかの条件を満たすレコードが  
取得されます

# テーブル構造

## 前提条件

- 原則的として、1つのテーブルには1つの意味になるようにする（≒第3正規形）

| 顧客ID | email   |
|------|---------|
| 1    | tanaka@ |
| 2    | sato@   |

顧客  
テーブル

| 製品ID | 名前      |
|------|---------|
| 1    | バスタオル   |
| 2    | フェイスタオル |

製品  
テーブル

| 取引ID | 顧客ID | 製品ID | 個数 |
|------|------|------|----|
| 1    | 1    | 1    | 4  |
| 2    | 1    | 2    | 4  |
| 3    | 2    | 1    | 10 |

取引  
テーブル

# テーブル構造

**consumer\_tbl** (顧客テーブル)

| consumer_id<br>顧客ID | name<br>名前 | email<br>メールアドレス | tel<br>電話番号 | address<br>住所 |
|---------------------|------------|------------------|-------------|---------------|
| 1                   | 千葉順        | chiba@hea        | 0801292XX   | 東京都世田         |
| 2                   | 室谷次郎       | muroya@h         | 044542XX    | 神奈川県川         |

↑ 同じ意味の値 ↓

**deal\_tbl** (取引テーブル)

| deal_id<br>取引ID | consumer_id<br>顧客ID | product_id<br>商品ID | count<br>個数 | delivery_flg<br>配送フラグ |
|-----------------|---------------------|--------------------|-------------|-----------------------|
| 1               | 1                   | 4                  | 10          | 0                     |
| 2               | 2                   | 3                  | 2           | 1                     |
| 3               | 2                   | 4                  | 1           | 0                     |

↑ 同じ意味の値 ↓

0 = 未配送  
1 = 配送済み

**product\_tbl** (商品テーブル)

| product_id<br>商品ID | product_name<br>商品名 | price<br>単価 |
|--------------------|---------------------|-------------|
| 3                  | バスタオル               | 4,400       |
| 4                  | フェイスタオル             | 1,100       |

# テーブル構造

- 正規形について
  - 下のような全部のデータが入ったテーブル 1 つ (非正規形と呼ぶ)でも良いのではないか？

| 取引ID | 顧客名 | email   | 製品名     | 単価   | 個数 | 金額    |
|------|-----|---------|---------|------|----|-------|
| 1    | 田中  | tanaka@ | バスタオル   | 4000 | 4  | 16000 |
| 2    | 佐藤  | sato@   | フェイスタオル | 2000 | 2  | 4000  |
| 3    | 千葉  | chiba@  | バスタオル   | 4000 | 2  | 8000  |
| 4    | 田中  | tanaka@ | フェイスタオル | 2000 | 1  | 2000  |

- 同姓同名がいたらどうする？
- 製品の名称が変わったら？



- 正規形について
  - 興味ある人は自分で調べてみてください。

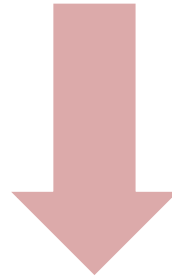
## 参考情報

- ・ 第1正規形 ⇒ 導出属性の削除と繰り返し項目の分離
- ・ 第2正規形 ⇒ 非キー依存の情報を分離
- ・ 第3正規形 ⇒ 主キーに依存しているが、  
情報として独立できるものを分離

参考 : [https://blog.codecamp.jp/php\\_normalization02](https://blog.codecamp.jp/php_normalization02)

## 問題その2.

新たに「メガバスタオル」  
7000円を取り扱うことになった。  
どのようなSQLを書けばよいか？



```
INSERT product_tbl (product_name, price)
VALUES ('メガバスタオル', 7000)
```

- あれ？ product\_idは書かなくていいの？
  - product\_idはproduct\_tblの **primary key** かつ、**auto increment** に設定されているため、不要です。
- **Primary Key**(主キー)とは・・・
  - テーブルにおいて一意に識別するための項目のこと。
    - ✓学生で言えば学籍番号。
    - ✓これによって同姓同名がいても識別できる。
- **Auto Increment**とは・・・
  - データを追加した時にカラムに対して現在格納されている最大の数値に 1 を追加した数値を自動で格納してくれます。

# テーブル構造

- phpMyAdminからの確認方法
  - テーブル選択 > 構造タブ から確認できます

The screenshot shows the 'Table Structure' tab in phpMyAdmin. The table 'product' has three columns: 'product\_id' (int(10), PRIMARY, AUTO\_INCREMENT), 'product\_name' (varchar(100), utf8\_general\_ci), and 'price' (int(10), utf8\_general\_ci). Annotations highlight the primary key icon on 'product\_id' and the 'AUTO\_INCREMENT' setting. Below the table structure, the 'Indexes' tab is selected, showing a PRIMARY index on the 'product\_id' column. An annotation points to this column in the index table.

| # | 名前           | データ型         | 照合順序            | 属性 | NULL | デフォルト値 | コメント | その他            | 操作        |
|---|--------------|--------------|-----------------|----|------|--------|------|----------------|-----------|
| 1 | product_id   | int(10)      |                 |    | いいえ  | なし     |      | AUTO_INCREMENT | 変更 削除 その他 |
| 2 | product_name | varchar(100) | utf8_general_ci |    | いいえ  | なし     |      |                | 変更 削除 その他 |
| 3 | price        | int(10)      | utf8_general_ci |    | いいえ  | なし     |      |                | 変更 削除 その他 |

Primary keyの目印

Auto Incrementの設定

☐ すべてチェックする    チェックしたものを: ☐ 表示    ☐ 変更    ☐ 削除    ☒ 主    ☐ ユニーク    ☐ インデックス    ☐ 全文

印刷    テーブル構造を確認する    カラムを移動させる    正規化

1 個のカラムを追加する    price の後へ    実行

インデックス

| 操作    | キー名     | データ型  | ユニーク | 圧縮  | カラム        | 一意な値の数 | 照合順序 | NULL | コメント |
|-------|---------|-------|------|-----|------------|--------|------|------|------|
| 編集 削除 | PRIMARY | BTREE | はい   | いいえ | product_id | 11     | A    | いいえ  |      |

Primary keyのカラム

## 問題その3.

product\_id が2 の製品の金額を  
4800円に変更したい。  
どのようなSQLを書けばよいか？



```
UPDATE product_tbl  
SET price = 4800  
WHERE product_id = 2
```

### ● レコードの更新 内容の変更

変更する

どのテーブルを

```
UPDATE product_tbl  
SET price = 4800  
WHERE product_id = 2
```

どのカラムを  
どんなデータに

どのデータだけにする？

### ● レコードの削除 製品の削除

削除する

どのテーブルを

```
DELETE from product_tbl  
WHERE product_id= 11
```

どのデータを

- WHERE句を書き忘れたらどーなるか？

---

# その他

---



## 問題その4.

現在取り扱っている製品数を知りたい

と言ってきた。

どのようなSQLを書けばよいか？



何行のデータがテーブルにあるか？ということ

- 製品数を算出しよう

```
SELECT count(*) FROM product_tbl
```

- 行数のカウントを数えているイメージ

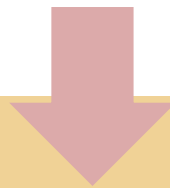
## 練習問題

単価が2000円以上、5000円以下の

製品を知りたい

をしたいと言ってきた。

どのようなSQLを書けばよいか？



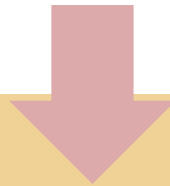
SQL Between で検索！

## 練習問題

製品名に「バスタオル」とつく製品

だけを表示したいと言ってきた。

どのようなSQLを書けばよいか？



SQL LIKE で検索！

# データベース設計

これまでは既に用意されたデータベースに対してSQLを用いてデータを取得、挿入、更新、削除等の処理を行って来ました。  
ここからはデータベースを自分で設計するときに必要な知識を学んでいきましょう。

## 問題その5.

顧客テーブル(consumer\_tbl)には他に  
どんな情報が必要だと考えられますか？



チャットに書いてみてください

## 問題その6.

顧客の性別をデータベースで  
管理したいと言われた。  
どうしたらよいか？



データベースを変更しなければならない！  
どのテーブルを変更すべきか？

# データベースの変更

## ● 新たに項目（カラム）を増やす(1/2)

- 管理したい項目を増やすにはデータベースのカラムを増やさなければいけません

Step①  
変更したいテーブルを選択

Step②  
「構造」タブをクリック

|   | 型        | 照合順序         | 属性              | NULL | デフォルト値 | コメント | その他            | 操作        |
|---|----------|--------------|-----------------|------|--------|------|----------------|-----------|
| 1 |          |              |                 | いいえ  | なし     |      | AUTO_INCREMENT | 変更 削除 その他 |
| 2 |          |              |                 | いいえ  | なし     |      |                | 変更 削除 その他 |
| 3 | email    | varchar(100) | utf8_general_ci | いいえ  | なし     |      |                | 変更 削除 その他 |
| 4 | tel      | varchar(11)  | utf8_general_ci | いいえ  | なし     |      |                | 変更 削除 その他 |
| 5 | address  | varchar(200) | utf8_general_ci | いいえ  | なし     |      |                | 変更 削除 その他 |
| 6 | birthday | date         |                 | いいえ  | なし     |      |                | 変更 削除 その他 |

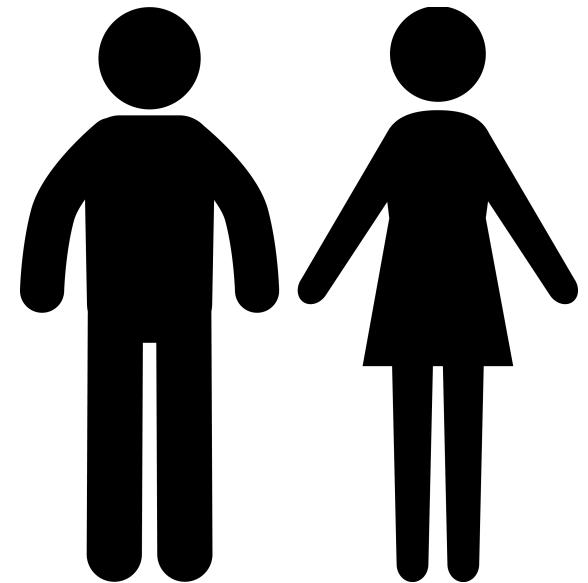
1 個のカラムを追加する birthday の後へ 実行

Step③  
カラムの追加を実行



## ● 性別についての考え方

- 男性/女性 という区別でよいのだろうか？
- 答えたくない人もいるだろうか？
- そもそも、聞く必要があるだろうか？



### ● 想定力とは

- 本日のプログラムでは下記をイメージしています

#### ① 業務における想定力

利用するユーザーについての理解

#### ② システムにおける想定力

業務を実現するためのシステム構成  
(+  $\alpha$  業務上は本来発生し得ないケースへの対応)

## ● 新たに項目（カラム）を増やす(2/2)

- カラムを追加するには  
「カラム名」、「データ型」、「長さ」  
の3つの情報を決める必要があります。

| 名前     | データ型 | 長さ/値 | デフォルト値 | 照合順序 | 属性 |
|--------|------|------|--------|------|----|
| gender | INT  | 2    | なし     |      |    |

構造

SQLのプレビュー 保存する

Step①  
カラム名の  
決定

Step②  
データ型を  
選択

Step③  
長さを決定

# データベースについて

## ● データ型について

- 各カラムにはデータ型とよばれる「**型**」があります。



The screenshot shows a database management interface for creating a table named 'tweet\_tbl'. It displays four columns: 'tweet\_id', 'account', 'contents', and 'input\_datetime'. A red box highlights the 'データ型' (Data Type) column, which contains dropdown menus with the following values: 'INT' for 'tweet\_id', 'VARCHAR' for 'account', 'VARCHAR' for 'contents', and 'DATETIME' for 'input\_datetime'.

| 名前             | データ型     | 長さ/値 | デフォルト |
|----------------|----------|------|-------|
| tweet_id       | INT      |      | なし    |
| account        | VARCHAR  |      | なし    |
| contents       | VARCHAR  |      | なし    |
| input_datetime | DATETIME |      | なし    |

- よく使うデータ型
- **int型** — 数値を保存 ex. 12345
- **varchar型** — 文字を保存 ex. おはよう
- **date型** — 日付時刻を保存  
✓ ex. 2014-07-07

# データベースについて

## ● データの長さについて

- 各カラムには保存できるデータの長さがあります。

| 名前             | データ型     | 長さ/値 | デフォルト値 | 照合順序            |
|----------------|----------|------|--------|-----------------|
| tweet_id       | INT      | 10   | なし     |                 |
| account        | VARCHAR  | 20   | なし     | utf8_general_ci |
| contents       | VARCHAR  | 140  | なし     | utf8_general_ci |
| input_datetime | DATETIME |      | なし     |                 |

文字化け防止のため！

- **int型** — 入力できるデータの**桁数**
  - ✓ ex. 2なら2桁の数値まで入る。
- **varchar型** — 入力できるデータの**文字数**
  - ✓ ex. 50なら50文字まで。

## 問題その7.

ポイント管理をしたい。

どのようなテーブルをつくるべきか？



最低限、誰が、何ポイント保有しているかを管理できないといけない。

# テーブルの作成

- 新たにテーブルを作成する
  - テーブル名「point\_tbl」とします(任意)
    - ✓ tbl は table = 表 の略

The screenshot shows a database management tool interface for creating a new table. The table name is 'point\_tbl'. The interface includes a table structure editor with columns and their properties, and a section for table options like storage engine and partitioning.

**Step①**  
新規作成をクリック

**Step②**  
テーブル名を記入

| 名前 | 長さ/値 | デフォルト値 | 照合順序 | 属性 | NULL                     | インデックス  |
|----|------|--------|------|----|--------------------------|---------|
| co | 10   | なし     |      |    | <input type="checkbox"/> | PRIMARY |
| po | 5    | なし     |      |    | <input type="checkbox"/> | ---     |
|    |      | なし     |      |    | <input type="checkbox"/> | ---     |
|    | INT  | なし     |      |    | <input type="checkbox"/> | ---     |

構造

テーブルのコメント: 照合順序: ストレージエンジン: InnoDB

PARTITION 定義:

パーティションによって: ( 式またはカラムリスト )

パーティション:

SQLのプレビュー 保存する

# テーブルの作成

## ● 新たにテーブルを作成する

| 名前          | データ型 | 長さ/値 | デフォルト値 | 照合順序 | 属性 | NULL                     | インデックス  |
|-------------|------|------|--------|------|----|--------------------------|---------|
| consumer_id | INT  | 10   | なし     |      |    | <input type="checkbox"/> | PRIMARY |
| point       | INT  | 5    | なし     |      |    | <input type="checkbox"/> |         |
|             |      |      | なし     |      |    | <input type="checkbox"/> |         |
|             |      |      | なし     |      |    | <input type="checkbox"/> |         |

Step③  
カラム名

Step④  
データ型

Step⑤  
長さ

Step⑥  
キー

ストレージエンジン: InnoDB

PARTITION 定義:

パーティションによって: ( 式またはカラムリスト )

パーティション:

SQLのプレビュー 保存する



**ポイント管理が可能な  
データベースを  
みんなで考えてみよう！**

### ● 想定力とは

- 本日のプログラムでは下記をイメージしています

#### ① 業務における想定力

利用するユーザーについての理解

#### ② システムにおける想定力

業務を実現するためのシステム構成  
(+  $\alpha$  業務上は本来発生し得ないケースへの対応)

---

# 課題タイムについて

---

### ● 課題タイム

- 個人で全9問 +  $\alpha$ の課題に取り組みます。
  - ✓ SQLの構文を調べれば解ける課題
  - ✓ 業務的な想定力を必要とする課題
  - ✓ テーブルの変更や新規作成が必要な課題
  - ✓ + $\alpha$ **任意**：アドバンス課題
    - 早くできた人向け
- 終了時間は16:20まで

## ●進め方

- 1人1人の適正を測るために以下の進め方を取ります。
- 原則として(午後は)個人ワーク
- 講師陣は答えを教えません
  - ✓ ヒントや思考を深めるお手伝い
- 課題 1 - 3 が終わったら提出、
- 課題 4 - 6 が終わったら提出
- 課題 7 - 9 が終わったら提出