

リスクリング

第5回 RDB MySQL / 全7回

RDB とは？

表計算のようなデータ構造でデータを扱う集積所みたいなところ。

社員番号	氏名	生年月日	部署
1	Tam	2006-01-01	雑務課
2	馬場	2003-09-08	秘書課
3	水尾	1980-08-29	総務課
4	多田	1990-09-12	電算課

- 上のようなデータ構造の単位を「テーブル」と呼びます。
- テーブルを複数組み合わせた構造の単位を「データベース」と呼びます。

MySQL とは？

- データベースを扱うオープンソースのソフトウェアです。
- いくつかあるデータベースソフトの中でも、有名所の一つです。

今日やること

- データベースを操作するには SQL という言語を使います。
- SQL には各種方言が存在しますが、基本的な扱いはほぼ同じなので、皆さんには MySQL で SQL の書き方を習得してもらいます。
- 方言に注意すれば、他のデータベースソフトも扱えるようになると思いますので、頑張ってください。
- なお、ライブラリ任せにしてしまって、SQL をまったく書かないという選択肢も可能ですが、書けたほうがバグの少ない、より効率の良いプログラムを作成することが容易になります。

(ここにツールの立ち上げ方を記述)

データベースの一覧

以下のようにすると、保存されているデータベースの一覧が表示されます。

```
mysql> SHOW DATABASES; ← 最後の「;（セミコロンと呼びます）」を忘れないように！
```

以下のようにして、普段利用するデフォルトデータベースを切り替えましょう。

```
mysql> USE データベース名;
```

テーブルの取得

以下のようにすると、データベースに登録されているテーブル一覧が表示されます。

```
mysql> SHOW TABLES;
```

注意点：

- くどいですが、最後の「; (セミコロン)」を忘れやすいので、気をつけましょう。
- アルファベットの大文字、小文字の区別はまったくありません。どちらで入力しても、おなじコマンドとして受け付けられるので、大文字、小文字の区別がある別名のテーブルなどは作成できません。

データベースの作成

今登録されているデータベースの一覧は以下のコマンドで取得できました。

```
SHOW DATABASES;
```

自分独自のデータベースを作成してみましょう。

```
CREATE DATABASE データベース名;
```

作成できたら、データベースの一覧を取得して、正常に作成されているか確認しましょう。

```
SHOW DATABASES;
```

エラーメッセージ

すでに存在するデータベースと同じ名前のデータベースを作成しようとするエラーになります。

```
ERROR 1007 (HY000): Can't create database 'hogehoge'; database exists
```

エラーメッセージは基本英語です。

Google 翻訳等を使ってもいいので、極力読んでいくようにしましょう。

バグの原因など、重要なメッセージが英語で出ていることが多いです。

テーブルの表示

`school` というデータベースを作成したとして、その中に格納されているテーブルの一覧を取得します。

```
CREATE DATABASE school;  
USE school;  
SHOW TABLES;
```

- ← データベース `school` を作成
- ← デフォルトデータベースを `school` に変更
- ← テーブルの一覧を取得

作ったばかりのデータベースにはテーブルが存在しないので、何も表示されないと思います。

データベースの削除

以下のコマンドで、今作ったばかりの school データベースが削除されます。

```
DROP DATABASE school;  
SHOW DATABASES;
```

← データベース一覧を取得して確認しましょう。

ラピュタより抜粋

滅びのまじない。いいまじないに力を与えるには、悪い言葉も知らなければいけないって。でも決して使うなって…。教わったとき、怖くて眠れなかった…。

テーブルの作成

データベースの作成からテーブルの作成までやってみましょう。

```
CREATE DATABASE school;  
USE school;  
CREATE TABLE student(  
    id        INT PRIMARY KEY,  
    name      VARCHAR(128),  
    grade     INT  
);
```

これにより、 **student** というテーブルが作成され、中身が **id** と **name** と **grade** になります。

データの追加

出来たばかりのテーブルにデータを追加してみましょう。

```
INSERT INTO student (id, name, grade) VALUES (1001, 'Tam', 1);  
INSERT INTO student (id, name, grade) VALUES (2001, '馬場', 2);  
INSERT INTO student (id, name, grade) VALUES (3001, '水尾', 3);  
INSERT INTO student (id, name, grade) VALUES (3002, '多田', 3);
```

コピーを活用して効率よく入力してください。

ただし、これらはたくさん使っていきますので、書き方は覚えていってください。

データの確認

今入力したばかりのデータを確認してみましょう。

```
SELECT * FROM student;
```

無事表示されましたでしょうか？

以下のようにすると、特定の列のみ取得することが可能です。

```
SELECT id, name FROM student;
```

テーブルの削除

以下のようにすると、テーブルの削除が出来ます。

```
DROP TABLE テーブル名;
```

再掲

滅びのまじない。いいまじないに力を与えるには、悪い言葉も知らなければいけないって。でも決して使うなって...。教わったとき、怖くて眠れなかった...

英語圏（スパイダーマン）

大いなる力には、大いなる責任が伴う。

並び替え

以下のようにすると、カラム名 id で降順に並び替えた結果が得られます。

```
SELECT id, name FROM student ORDER BY id DESC;
```

昇順にするためには、「DESC」の代わりに「ASC」が使えます。

検索

以下のようにすると、条件に合致したレコードのみを取り出すことができます。

ここでは、学年（grade）が2年以下を取り出しています。

```
SELECT * FROM student WHERE grade <= 2;
```


更新

データの更新は以下のようになります。

```
UPDATE student SET name = 'Mizuo' WHERE id = 3001;
```

これにより、id が 3001番の「水尾」が「Mizuo」に変更されます。

トランザクション

複数の人が異なるデータを書き込もうとすると困った事態になります。

この問題を回避するためには、以下のようにします。

```
START TRANSACTION;  
UPDATE student SET name = 'MIZUO' WHERE id = 3001;  
COMMIT; ←ここまでエラーがなければ、書き込む  
or  
ROLLBACK; ←エラーがあれば、変更をキャンセルして元に戻す。
```

Node.js から MySQL への接続

```
% npm install mysql
```

```
const mysql = require('mysql');  
  
const connection = mysql.createConnection({  
  host: "localhost",  
  user: "【MySQLユーザー名】",  
  password: "【MySQLパスワード】",  
  database: "【データベース名】"  
});
```

MySQL の操作

```
connection.connect(function(err) {  
  if (err) throw err;  
  console.log("Connected to MySQL DB!");  
  
  /// 例 : テーブルの全レコード取得  
  const sql = `  
    SELECT * FROM student;  
  `;  
  
  connection.query(sql, function (err, result) {  
    if (err) throw err;  
    /// 全レコード表示  
    console.log("Result: ", result);  
  });  
});
```

演習

なにかの CSV をテーブルに書き込む Node.js ?