1. webシステム基礎実験（JavaScriptクラス）本格的なDBの使い方編

今回は，テーブルを連結して，多様なデータを扱うことを目標とする．

1.1. 準備

いつもどおりPaiza.cloudを使用する． サーバを起動する際，Node.jsとMySQLを使用するようにチェックを入れること．

起動したらターミナルから以下のように順番に入力して，必要なデータをセットしよう．

まずはリポジトリのcloneを行う．

```
~$ git clone https://github.com/sudahiroshi/websystem3.git
Cloning into 'websystem3'...
remote: Enumerating objects: 12, done.
create table player ( id int auto_increment not null primary key, name
varchar(1remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 12 (delta 1), reused 9 (delta 1), pack-reused 0
Unpacking objects: 100% (12/12), done.
~$
```

ディレクトリを変更する．

```
~$ cd websystem3
~/websystem3$
```

DBの初期化を行う．

```
~/websystem3$ sudo mysql < init.sql
~/websystem3$
```

データの流し込みを行う． このとき，Warningが発生しているが，今は気にしなくて良い．

```
~/websystem3$ mysql -u node -pwebsystem web < db.sql
mysql: [Warning] Using a password on the command line interface can be
insecure.
~/websystem3$
```

1.2. DBの確認

ここまできちんとできていることを，データを閲覧して確認してみよう． 今回例として使用したのは日本プロ野球の打撃成績である． 年度ごとの打撃成績，選手名，球団名をそれぞれテーブルとしている． なお，データは日本野球機構の個人年度別成績より抜粋した．

まずはmysqlコマンドの起動．

```
~/websystem3$ mysql -u node -pwebsystem web
mysql: [Warning] Using a password on the command line interface can be
insecure.
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.7.24-0ubuntu0.18.04.1 (Ubuntu)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights
reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.

mysql>
```

続いて，選手の一覧． 昔の選手が多いのは気にしないでください．

```
mysql> select * from player;
+----+--------------+
| id | name         |
+----+--------------+
|  1 | イチロー     |
|  2 | 秋山幸二     |
|  3 | 落合博満     |
+----+--------------+
3 rows in set (0.00 sec)

mysql>
```

続いて，球団一覧． 上記の選手が所属したことのある球団のみ登録してあります．

```
mysql> select * from team;
+----+--------------------+
| id | name               |
+----+--------------------+
```

```
|  1 | オリックス          |
|  2 | 西武                |
|  3 | 福岡ダイエー        |
|  4 | ロッテ              |
|  5 | 中日                |
|  6 | 読売                |
|  7 | 日本ハム            |
+----+--------------------+
7 rows in set (0.00 sec)

mysql>
```

続いて打撃成績.　ここでは最初の10個分のデータのみ表示しています.

```
mysql> select * from batting limit 10;
+----+------+-----------+---------+------+------+------+------+------+
| id | year | player_id | team_id | PA   | AB   | H    | HR   | R    |
+----+------+-----------+---------+------+------+------+------+------+
|  1 | 1992 |         1 |       1 |   99 |   95 |   24 |    0 |    5 |
|  2 | 1993 |         1 |       1 |   67 |   64 |   12 |    1 |    3 |
|  3 | 1994 |         1 |       1 |  616 |  546 |  210 |   13 |   54 |
|  4 | 1995 |         1 |       1 |  613 |  524 |  179 |   25 |   80 |
|  5 | 1996 |         1 |       1 |  611 |  542 |  193 |   16 |   84 |
|  6 | 1997 |         1 |       1 |  607 |  536 |  185 |   17 |   91 |
|  7 | 1998 |         1 |       1 |  558 |  506 |  181 |   13 |   71 |
|  8 | 1999 |         1 |       1 |  468 |  411 |  141 |   21 |   68 |
|  9 | 2000 |         1 |       1 |  459 |  395 |  153 |   12 |   73 |
| 10 | 1981 |         2 |       2 |    6 |    5 |    1 |    0 |    0 |
+----+------+-----------+---------+------+------+------+------+------+
10 rows in set (0.00 sec)

mysql>
```

battingの項目は以下の通り.

項目名	意味
id	ID
yser	年度
player_id	上の方に書いた選手テーブルのid
team_id	上の方に書いた球団テーブルのid
PA	打席数
AB	打数
H	安打数
HR	ホームラン数

項目名	意味
R	打点

1.3. 表の結合

このままでは，とても見づらい出力結果になっているので，複数のテーブルを結合して見やすくしよう．

まずは，選手名をきちんと表示させる例を示す． ここで，`inner join`が表の結合を行うための単語である． SQLにおいて，表の結合はいくつかの種類があり，最もよく使うのが`inner join`である． 使い方であるが，`from batting inner join player`のように`from`の後ろにメインとなるテーブル名を書き，その後に`innserjoin`と結合するテーブルを記述する．

そして，最も重要なのがその後にある`on`の項目である． この例では`on batting.player_id = player.id`となっている． これは，`batting`テーブルの`player_id`と`player`テーブルの`id`が同じになるように結合することを意味している．

また，`select`の後の，表示項目欄にテーブル名が付いている． これは，複数のテーブルからデータを取得するので，テーブル名がないとどの項目かわからなくなってしまうのを防ぐためである．

```
mysql> select batting.id, batting.year, batting.HR, player.name
 from batting
 inner join player on batting.player_id = player.id
 limit 10;
+----+------+------+--------------+
| id | year | HR   | name         |
+----+------+------+--------------+
|  1 | 1992 |    0 | イチロー     |
|  2 | 1993 |    1 | イチロー     |
|  3 | 1994 |   13 | イチロー     |
|  4 | 1995 |   25 | イチロー     |
|  5 | 1996 |   16 | イチロー     |
|  6 | 1997 |   17 | イチロー     |
|  7 | 1998 |   13 | イチロー     |
|  8 | 1999 |   21 | イチロー     |
|  9 | 2000 |   12 | イチロー     |
| 10 | 1981 |    0 | 秋山幸二     |
+----+------+------+--------------+
10 rows in set (0.00 sec)

mysql>
```

1.4. 練習問題

上の例のように，打撃成績のid，年度，ホームラン数と球団名を表示するqueryを記述せよ． なお，選手名は不要である．

1.5. 3つの表の結合

続いて，3つのテーブルを結合してみよう． その場合，`inner join`と`on`の組が増えることとなる． 具体的な例を以下に示す．

```
mysql> select batting.id, batting.year, batting.HR, player.name, team.name
 from batting
 inner join player on batting.player_id = player.id
 inner join team on batting.team_id = team.id
 limit 10;
+----+------+------+--------------+-----------------+
| id | year | HR   | name         | name            |
+----+------+------+--------------+-----------------+
|  1 | 1992 |    0 | イチロー     | オリックス      |
|  2 | 1993 |    1 | イチロー     | オリックス      |
|  3 | 1994 |   13 | イチロー     | オリックス      |
|  4 | 1995 |   25 | イチロー     | オリックス      |
|  5 | 1996 |   16 | イチロー     | オリックス      |
|  6 | 1997 |   17 | イチロー     | オリックス      |
|  7 | 1998 |   13 | イチロー     | オリックス      |
|  8 | 1999 |   21 | イチロー     | オリックス      |
|  9 | 2000 |   12 | イチロー     | オリックス      |
| 10 | 1981 |    0 | 秋山幸二     | 西武            |
+----+------+------+--------------+-----------------+
10 rows in set (0.00 sec)

mysql>
```

1.6. 集計

SQLには，最大値を取得する関数が備わっている． 使用例を以下に示す． ここでは，ホームラン数（項目名：HR）の最大値を調べている．

```
mysql> select max(HR) from batting;
+---------+
| max(HR) |
+---------+
|      52 |
+---------+
1 row in set (0.00 sec)

mysql>
```

さて，年間52本のホームラン数を記録したのは誰でしょう？ これを調べるためには，まだまだSQLの詳しい使い方を知る必要があるので，後回しにします．

続いて，playerごとの最大ホームラン数を調べてみましょう． ここで出てくる単語が`group by`です． 集計時に`group by`の後ろにまとめたい項目名を書いておきます． ここでは，`player_id`ごとに最大値を求めるというqueryになります．

```
mysql> select player_id,max(HR) from batting group by player_id;
+-----------+---------+
| player_id | max(HR) |
+-----------+---------+
|         1 |      25 |
|         2 |      43 |
|         3 |      52 |
+-----------+---------+
3 rows in set (0.00 sec)

mysql>
```

1.7. 練習問題

球団ごとの最大ホームラン数を調べるqueryを記述せよ． なお，選手名も球団名も不要で，IDが表示されれば良い.

1.8. おまけ

選手名と，年間ホームラン数を表示する場合は以下のようなqueryを組めば良い.

```
mysql> select player.name, max(batting.HR)  from batting inner join player
on ( batting.player_id = player.id ) group by batting.player_id;
+--------------+-----------------+
| name         | max(batting.HR) |
+--------------+-----------------+
| イチロー     |              25 |
| 秋山幸二     |              43 |
| 落合博満     |              52 |
+--------------+-----------------+
3 rows in set (0.01 sec)

mysql>
```

ちょっと長くて入力しづらいので，以下のようにasを用いて短くする例を示す． ここで，pはplayerの別名を，bはbattingの別名を示す.

```
mysql> select p.name, max(b.HR) from batting as b inner join player as p
on ( b.player_id = p.id ) group by b.player_id;
+--------------+-----------+
| name         | max(b.HR) |
+--------------+-----------+
| イチロー     |        25 |
| 秋山幸二     |        43 |
| 落合博満     |        52 |
+--------------+-----------+
3 rows in set (0.00 sec)
```

```
mysql>
```

どちらの表記でも良いが，後々プログラムから使用する際に項目名が`max(batting.HR)`のように長いと使いづらいので，そこだけは別名を定義するほうが良い．

```
mysql> select p.name, max(b.HR) as HR from batting as b inner join player
as p on ( b.player_id = p.id ) group by b.player_id;
+--------------+------+
| name         | HR   |
+--------------+------+
| イチロー     |   25 |
| 秋山幸二     |   43 |
| 落合博満     |   52 |
+--------------+------+
3 rows in set (0.00 sec)

mysql>
```

1.9. データの追加

以前，データベースの基本操作としてCRUD（Create：作成， Read：読み出し， Update：更新，Delete：削除）を紹介した． すでにReadはできているので，次にCreateを試してみよう． 具体的には，Playerとbattingにデータを追加するためにinsertを使用する．

insertの使い方であるが`insert into <テーブル名> ( 項目名1, 項目名2, ... ) values ( 値1, 値2, ... );`のように記述する． ここで，`<テーブル名>`や`項目1`などは，具体的な文字を入れるので，このまま入力しないこと．

1.9.1. 追加するデータ

ここでは，現在メジャーリーグで活躍している大谷翔平の打撃成績を例に取る． そのため，player，team，battingという3つのテーブルに順番にデータを追加する．

まずは簡単なテーブルである`player`にデータを追加してみよう．

```
mysql> insert into player ( name ) values ( '大谷翔平' );
Query OK, 1 row affected (0.01 sec)

mysql>
```

ちゃんと登録できているか確認しよう． 以下のように表示されればOKである． ちなみに，insertの中ではidに関する記述がない． これは，テーブルを作成するときに，idを自動的に割り振るようにしているためである．

```

```

```
mysql> select * from player;
+----+--------------+
| id | name         |
+----+--------------+
|  1 | イチロー     |
|  2 | 秋山幸二     |
|  3 | 落合博満     |
|  4 | 大谷翔平     |
+----+--------------+
4 rows in set (0.00 sec)

mysql>
```

続いて，teamに登録しよう． すでに日本ハムが登録されているが，現在の球団名は北海道日本ハムなので，それに合わせて新しい球団を登録する．

```
mysql> insert into team ( name ) values ( '北海道日本ハム' );
Query OK, 1 row affected (0.00 sec)

mysql>
```

以下，確認．

```
mysql> select * from team;
+----+----------------------+
| id | name                 |
+----+----------------------+
|  1 | オリックス           |
|  2 | 西武                 |
|  3 | 福岡ダイエー         |
|  4 | ロッテ               |
|  5 | 中日                 |
|  6 | 読売                 |
|  7 | 日本ハム             |
|  8 | 北海道日本ハム       |
+----+----------------------+
8 rows in set (0.00 sec)

mysql>
```

それでは，年間成績を追加していこう． 大谷翔平の年間成績を以下の表に示す．

年度	打席数	打数	安打	ホームラン	打点
2013	204	189	45	3	20
2014	234	212	58	10	31

年度	打席数	打数	安打	ホームラン	打点
2015	119	109	22	5	17
2016	382	323	104	22	67
2017	231	202	67	8	31

これらのデータを1度に追加することはできないので，1年分ずつ登録していく． 例えば2013年度のデータを追加する場合，以下のようにすれば良い．

```
mysql> insert into batting ( year, player_id, team_id, PA, AB, H, HR, R )
values ( 2013, 4, 8, 204, 189, 45, 3, 20 );
Query OK, 1 row affected (0.00 sec)

mysql>
```

続いて確認であるが，全てのデータが表示されると画面が埋まってしまうので，以下のようにwhereを使って，条件を指定する． この例ではwhereの後ろに条件としてplayer_id=4と書かれているので，大谷翔平のデータのみが表示される．

```
mysql> select * from batting where player_id=4;
+----+------+-----------+---------+------+------+------+------+------+
| id | year | player_id | team_id | PA   | AB   | H    | HR   | R    |
+----+------+-----------+---------+------+------+------+------+------+
| 50 | 2013 |         4 |       8 |  204 |  189 |   45 |    3 |   20 |
+----+------+-----------+---------+------+------+------+------+------+
1 row in set (0.01 sec)

mysql>
```

1.9.2. 練習問題

selectを使用して，大谷翔平の打撃成績を表示せよ． このとき，選手名と球団名も表示するようにせよ．

ヒント：inner joinとwhereを組み合わせる．

1.10. データの削除

追加ができたと思うので，続いてデータを削除してみよう． 削除するためにはdeleteを使用するが，重要な点として，whereを用いて削除するデータを指定しないと，全てのデータが削除される点である． 書式としてはdelete from <テーブル名> where 条件である．

上で追加した大谷翔平の2013年度のデータを削除する例を示す． ここで，対象とするデータの指定方法として，idを使用する． なお，player_idとyearを組み合わせても良いが，条件指定を誤るとたくさんのデータが削除される恐れがあるため，idの使用を勧める．

```
mysql> delete from batting where id=50;
Query OK, 1 row affected (0.01 sec)

mysql>
```

以下のようにして，データが残っていないことを確認しよう．

```
mysql> select * from batting where player_id=4;
Empty set (0.00 sec)

mysql>
```

1.10.1. 練習問題

大谷翔平の全年度のデータを追加し，最新の年度のデータのみを削除してみよう．

1.11. データの更新

続いて，データを更新してみよう． 更新するためには`update`を使用する． こちらも`delete`と同様，`where`を用いて更新するデータを指定しなければ，全てのデータが更新される． 書式としては`update <テーブル名> set <項目名> = <値> where 条件`である． もし複数の項目を更新したければ，`update <テーブル名> set <項目名1> = <値1>, <項目名2> = <値2>, ... where 条件`のように記述することも可能である．

具体的な例を示す． まずは，大谷翔平の2013年度のデータを確認する．

```
mysql> select * from batting where player_id=4 and year=2013;
+----+------+-----------+---------+------+------+------+------+------+
| id | year | player_id | team_id | PA   | AB   | H    | HR   | R    |
+----+------+-----------+---------+------+------+------+------+------+
| 51 | 2013 |         4 |       8 |  204 |  189 |   45 |    3 |   20 |
+----+------+-----------+---------+------+------+------+------+------+
1 row in set (0.00 sec)

mysql>
```

idが51であることが確認できる． なお，deleteで削除したidは，欠番となる． 続いてidが51のデータのPAを500にしてみる．

```
mysql> update batting set PA=500 where id=51;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

確認する.

```
mysql> select * from batting where player_id=4 and year=2013;
+----+------+-----------+---------+------+------+------+------+------+
| id | year | player_id | team_id | PA   | AB   | H    | HR   | R    |
+----+------+-----------+---------+------+------+------+------+------+
| 51 | 2013 |         4 |       8 |  500 |  189 |   45 |    3 |   20 |
+----+------+-----------+---------+------+------+------+------+------+
1 row in set (0.00 sec)

mysql>
```

1.11.1. 練習問題

上記の変更を，再度updateを用いて元のデータに戻してみよう.

1.12. Webシステム化してみよう

1.12.1. 準備

データを追加・変更・削除するためのQueryを理解したところで，Webブラウザから追加するようにしてみよう． その前に，すぐに確認できるように，各データの一覧を取得できるページから開発しよう． すでに，データ一覧するページを作成したと思うが，ここでは改めて以下のURLでアクセスできるようにしよう． これらのページが有るのと無いのとでは，その後の作業効率に差が出てくる.

参考までに，URLによって様々なデータ形式にアクセスできるような仕組みをRESTfulと呼ぶ． RESTとはREpresentational State Transferの略で，大雑把に言うとクライアント・サーバ間で統一的にデータを扱うための階層的な設計のことである． 今は「？」状態で構わないが，`/players`などでアクセスするデータを明確にすると考えておいて欲しい.

URLの末尾	内容
/players	選手一覧
/teams	球団一覧
/battings	打撃成績一覧

1.12.2. 選手追加のためのページについて

以前，都道府県データベースの中で，検索条件をWebページから与える例を紹介した． 細かく言うと，その時はパラメータがなければ全て表示して，パラメータがあれば並び順を変えるなどを行った． また一覧ページにformが存在していた.

ここでは，もう少しシンプルにするために，以下のURLに分割して考える.

URLの末尾	動的・静的	内容

URLの末尾	動的・静的	内容
/playsers_new	静的	選手を追加するためのformのみのページ
/playsers_create	動的	上記formから呼び出されるページで，実際にデータベースにデータを追加する．追加したら追加しましたなどと表示する．

また，`/players`には`/players_new`へのリンクを，`/players_create`には`/players`へのリンクを付けておくと使い勝手が良い． それでは，この設計にしたがってファイルを作成していこう．