

# 計算機科学実験及び演習 4 データベース 報告書

成瀬 真

(学籍番号: 1029287040)

提出日: 2019 年 1 月 28 日

## 1 システム概要

本システムはカヌースプリント大会の運営システムである。筆者は全日本学生カヌー連盟の学生委員長であり、1 回生時から大会の運営に携わってきた。そんな大会でも、エントリーや抽選作業は未だに手作業である。大会運営がより円滑になることを考え、本システムを設計し、導入することを検討している。

本システムは 3 つのセクションからなる。

1 つ目は、選手登録セクションである。これは、各大学の代表者（主務など）が利用するシステムである。代表者は、選手情報、どのレースにエントリーを行うか、2 人乗り種目ではどの相手をペアにするかを登録する。

2 つ目は、大会運営セクションである。これは大会運営者が操作するものであり、いくつかの機能を備えている。まずは、大会前にスタートリストを作成する。システム上で、選手へのゼッケン番号の割り振り、レースの作成、抽選を行うことができる。大会中では結果を入力する。

3 つ目は、スタートリスト・速報セクションである。大会前では運営者が作成したスタートリストを確認することができ、大会中では運営者が入力した結果を即座に確認することができる。

## 2 実体関連図

以下に実体関連図を示す。実体「選手」は 1 人の選手を表す。実体「選手」は関連「エントリー」によって実体「レース」に関連している。実体「レース」は 1 つのレースを表す。また、実体「選手」は関連「ゼッケン割当」によって実体「ゼッケン」に関連している。これは、1 人のレースに 1 つのゼッケンを割り当てるという 1 対 1 の関連である。関連「出場」によって、実体「ゼッケン」、実体「レース」、実体「結果」は関連している。これは、あるレースにあるゼッケン番号の選手が〇〇レーンに出場していて、それに対して 1 つの結果が存在するという関連である。実体「結果」は 1 つの結果を表す。

関数従属性や挿入のパターンを考え、次章のように関係を構築した。

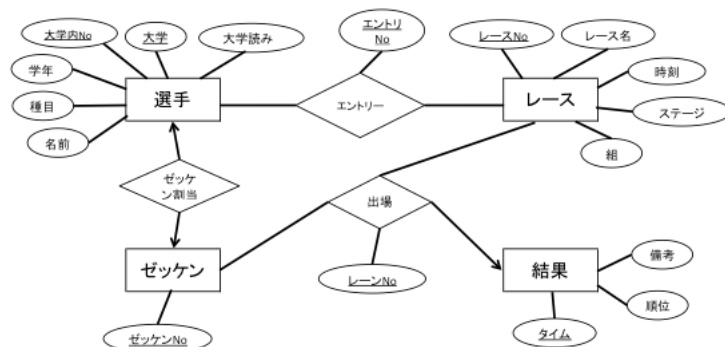


図 1 実体関連図.

### 3 関係スキーマ

#### 3.1 関係選手 (players)

この関係は 1 つの選手の情報を示している。以下に要素とデータの例を示す。*dom\_no* は大学内で仮に発行される通し番号である。*p\_name* は選手名である。*type* は種目であり、*K*(カヤック)、*C*(カナディアン)、*WK*(女子カヤック)、*JK*(ジュニアカヤック) などがある。*grade* は学年である。

| Column         | Type                  |
|----------------|-----------------------|
| univ_name(key) | character varying(30) |
| dom_no(key)    | integer               |
| p_name         | character varying(20) |
| type           | character varying(4)  |
| grade          | integer               |

| univ_name | dom_no | p_name | type | grade |
|-----------|--------|--------|------|-------|
| 鹿屋体育大学    | 1      | 佐藤 博幸  | C    | 3     |
| 鹿屋体育大学    | 2      | 森黒 開   | K    | 3     |
| 鹿屋体育大学    | 3      | 田原 瞭太  | K    | 3     |

#### 3.2 関係エントリー (entries)

関係エントリーはその選手がどのレースにエントリーするかを表す。*race\_name* が出場するレースの名前となる。

| Column      | Type                  |
|-------------|-----------------------|
| ent_no(key) | integer               |
| univ_name   | character varying(30) |
| dom_no      | integer               |
| race_name   | character varying(20) |

| ent_no | univ_name | dom_no | race_name |
|--------|-----------|--------|-----------|
| 597    | 関西学院大学    | 1      | K-1-1000m |
| 598    | 関西学院大学    | 1      | K-2-1000m |
| 599    | 関西学院大学    | 2      | K-1-1000m |

### 3.3 関係ペア (pairs)

関係ペアはある選手のペアがどの選手であるかを表している。ペアは大学内で組むものであるため、ペアの相手を大学内ナンバーで指定している。

| Column    | Type                  |
|-----------|-----------------------|
| univ_name | character varying(30) |
| dom_no    | integer               |
| pair_no   | integer               |

| univ_name | dom_no | pair_no |
|-----------|--------|---------|
| 京都大学      | 4      | 7       |
| 京都大学      | 7      | 4       |
| 京都大学      | 5      | 6       |

### 3.4 関係ゼッケン (bibs)

関係ゼッケンは、各大学の選手にゼッケン番号を割り振るものである。選手を大学順、学年順、種目順に並び替え、その順番にゼッケン番号を割り振る。大会期間中はゼッケン番号で選手を管理することになるため、以降はこのテーブルを介して選手情報を参照することになる。*bib\_no* がゼッケン番号を表す。

| Column    | Type                  |
|-----------|-----------------------|
| bib_no    | integer               |
| univ_name | character varying(30) |
| dom_no    | integer               |

| bib_no | univ_name | dom_no |
|--------|-----------|--------|
| 1      | 鹿屋体育大学    | 1      |
| 2      | 鹿屋体育大学    | 2      |
| 3      | 鹿屋体育大学    | 3      |

### 3.5 関係レース (races)

関係レースは1つのレースを表す。運営者はどのレースを行うかを決定し、タイムテーブルを作成し、レースナンバーと時間を入力する。*stage* は1次 *H*(一次予選)、*H*(予選)、*SF*(準決勝)、*F*(決勝) がある。セットは組数である。*Final* の組みは *null* となっている。大会によっては決勝を *A* 決勝と *B* 決勝で2組行う場合があるので今後対応する。

| Column    | Type                   |
|-----------|------------------------|
| race_no   | integer                |
| race_time | character varying(20)  |
| race_name | character varying(100) |
| stage     | character varying(4)   |
| set       | integer                |

| race_no | race_time | race_name | stage | set |
|---------|-----------|-----------|-------|-----|
| 1       | 9:35      | K-1-1000m | 1次H   | 1   |
| 2       | 9:40      | K-1-1000m | 1次H   | 2   |
| 3       | 9:50      | K-1-1000m | 1次H   | 3   |

### 3.6 関係出場 (participates)

関係出場は、そのレースにどの選手が出場するかを表す関係である。運営者はシステムを用いて抽選を行うことにより、このテーブルに出場情報が登録される。

| Column  | Type    |
|---------|---------|
| race_no | integer |
| rane_no | integer |
| bib_no  | integer |

| race_no | rane_no | bib_no |
|---------|---------|--------|
| 1       | 1       | 83     |
| 1       | 2       | 112    |
| 1       | 3       | 24     |

### 3.7 関係順位 (ranks)

関係順位はそのレースでのレーンごとの順位を表す。大会中に挿入する。

| Column  | Type    |
|---------|---------|
| race_no | integer |
| rane_no | integer |
| rank    | integer |

| race_no | rane_no | rank |
|---------|---------|------|
| 1       | 1       | 3    |
| 1       | 2       | 1    |
| 1       | 3       | 6    |

### 3.8 関係結果 (results)

関係順位はそのレースでのレーンごとのタイムを表す。大会中に挿入する。

| Column  | Type                      |
|---------|---------------------------|
| race_no | integer                   |
| rank    | integer                   |
| time    | time(3) without time zone |

| race_no | rank | time        |
|---------|------|-------------|
| 1       | 1    | 00:04:01.27 |
| 1       | 2    | 00:04:02.56 |
| 1       | 3    | 00:04:03.23 |

### 3.9 関係大学読みがな (reads)

この関係は各大学の読みがなを指定する。選手名簿を並び替える際、大学名を辞書順に並び替えるために大学名の呼び仮名が必要となった。

| Column    | Type                  |
|-----------|-----------------------|
| univ_name | character varying(30) |
| univ_read | character varying(30) |

| univ_name | univ_read  |
|-----------|------------|
| 京都大学      | きょうとだいがく   |
| 神戸大学      | こうべだいがく    |
| 立命館大学     | りつめいかんだいがく |

## 4 機能・インタフェース

第1章で述べた通り、このシステムには3つのセクションがある。各セクションでの主な画面とそこで行える機能を説明していく。

### 4.1 トップページ

トップページから特定の役割に応じたページへと進むことができる。各大学の代表者は選手登録へ、運営者は関係者用へ、その他の選手、一般の方々は記録速報へと進む。

#### 選手登録

選手登録・変更される大学代表者の方は[こちら](#)

#### 記録速報

記録速報は[こちら](#)で実施中です

#### 関係者用

[要パスワード](#)

図2 トップページ.

### 4.2 大学名入力画面

選手登録を行う際、はじめに大学名を入力する。簡単のため、この今のシステムでは大学名を入力しその大学名の選手データを検索するというシステムとなっているが、実際に運用するには大学ごとにパスワードを設定しログインするのが良いだろう。

## 大学名を入力してください

大学名:

図3 大学名入力画面.

### 4.3 選手登録画面

大学名を入力すると、選手登録画面へと進む。この画面では、自大学の選手情報の登録を行う。「現在の選手登録内容」では *SQL* 文の検索を利用し、自大学に該当するエントリーを表示している。もしも前画面で選手登録内容が存在しない大学名を入力するとここには何も登録されない。選手を追加する場合、画面下部にある「選手追加」を用いる。ここで入力された内容は表へと挿入される。入力しやすいように、氏名以外の項目はプルダウン形式にしている。また、表内に削除と修正のボタンがある。削除ボタンを押下するとその選手のデータは削除される。また、修正ボタンを押下すると選手情報修正画面へと遷移する。選手情報を登録したら、次のエントリー登録画面へと進む。

#### 選手登録・変更

大学名: 京都大学  
この内容でよければ

#### 現在の選手登録内容

| 大学内No | 選手名    | 種目 | 学年 | 操作  |
|-------|--------|----|----|---|
| 1     | 高尾 充政  | C  | 4  | <input type="button" value="削除"/> <input type="button" value="修正"/> |
| 2     | 田中 晴哉  | C  | 4  | <input type="button" value="削除"/> <input type="button" value="修正"/> |
| 3     | 三津 海童  | C  | 4  | <input type="button" value="削除"/> <input type="button" value="修正"/> |
| 4     | 池端 隆彦  | K  | 4  | <input type="button" value="削除"/> <input type="button" value="修正"/> |
| 5     | 市川 公惇  | K  | 4  | <input type="button" value="削除"/> <input type="button" value="修正"/> |
| 6     | 奥平 雅樹  | K  | 4  | <input type="button" value="削除"/> <input type="button" value="修正"/> |
| 7     | 村田 翔   | K  | 4  | <input type="button" value="削除"/> <input type="button" value="修正"/> |
| 8     | 大花 ちなみ | WK | 4  | <input type="button" value="削除"/> <input type="button" value="修正"/> |
| 9     | 原田 陽弓  | WK | 4  | <input type="button" value="削除"/> <input type="button" value="修正"/> |
| 10    | 小澤 佑人  | C  | 3  | <input type="button" value="削除"/> <input type="button" value="修正"/> |
|       |        |    |    | <input type="button" value="削除"/> <input type="button" value="修正"/> |

図4 選手登録画面.

#### 4.4 選手情報修正画面

この画面では選手情報の修正を行う。画面上部には現在の選手情報が表示されており、下部には修正するためのフォームが配置されている。フォームの初期値が現在の内容となっているため、変えたい項目のみ変えることができる。変更ボタンを押下することで、表に対して更新が行われる。



### 選手情報修正

**変更前**

大学名： 京都大学  
大学内ナンバー： 12  
選手名： 成瀬 真  
種目： C  
学年： 3

↓  
↓  
↓

**変更後**

大学名：   
選手名：   
種目：    
学年：  

大学名：  の選手登録に

図 5 選手情報修正画面。

#### 4.5 エントリー登録画面

この画面ではエントリーの登録、変更を行うことができる。一部の人を除いてほとんどの人がシングル種目・ペア種目に参加するため、一括登録ボタンを設けてある。このボタンを押下すると、全ての選手が、自分の種目に応じたシングルとペアのレースに参加することになる。また、一括削除を押すとその大学の全てのエントリーを削除する。また、このテーブルは *players* と *entries* を自然結合して選手のエントリーを表現している。手動削除・手動追加についても選手登録同様に行うことができる。また、更新についてはあまり需要が見込めないため実装していない。



## エントリー登録・変更

大学名：京都大学

選手登録に [戻る](#)

この内容でよければ [ペア登録へ](#)

## 現在のエントリー内容

[全員シングル・ペアに登録する](#)

[全て削除して初期化する](#)

| 大学内No | 選手名   | 種目 | エントリー     | 操作                 |
|-------|-------|----|-----------|--------------------|
| 1     | 高尾 充政 | C  | C-1-1000m | <a href="#">削除</a> |
| 1     | 高尾 充政 | C  | C-2-1000m | <a href="#">削除</a> |
| 2     | 田中 晴哉 | C  | C-1-1000m | <a href="#">削除</a> |
| 2     | 田中 晴哉 | C  | C-2-1000m | <a href="#">削除</a> |
| 3     | 三津 海童 | C  | C-1-1000m | <a href="#">削除</a> |
| 3     | 三津 海童 | C  | C-2-1000m | <a href="#">削除</a> |
| 4     | 池端 隆彦 | K  | K-1-1000m | <a href="#">削除</a> |

図 6 エントリー登録画面。

## 4.6 ペア登録画面

この画面では 2 人乗り種目の相方を決定する。相手の大学内ナンバーを入力することで相方を指定することができる。自分がある人をペアとして登録した場合、その人も自分をペアとして登録する。すなわち、2 回表に対して挿入が行われる。これを利用して、トランザクションを利用してエラー処理を行っている。すでに誰かとペアを組んでいる人をペアの相方として登録しようとした場合、自分は相手をペアとして指定することができるが、相手が自分をペアとして指定しようすると、既に誰かを登録済みのためキーエラーとなる。この場合、処理をロールバックしてエラーメッセージを返す。このようにして、1 人が複数の相手とペアを組めないようにしている。また、このエラー処理は自分をペアの相手として指定しようとした場合でもエラーを拾う。

### ペア登録・変更

大学名：京都大学

エントリー登録に [戻る](#)

この内容でよければ [登録内容の確認へ](#)

#### 現在のペア登録内容

| 大学内No | 選手名   | エントリー     | ペアNo | 操作                 |
|-------|-------|-----------|------|--------------------|
| 1     | 高尾 充政 | C-2-1000m | 2    | <a href="#">削除</a> |
| 2     | 田中 晴哉 | C-2-1000m | 1    | <a href="#">削除</a> |
| 3     | 三津 海童 | C-2-1000m | 19   | <a href="#">削除</a> |
| 10    | 小澤 佑人 | C-2-1000m | 11   | <a href="#">削除</a> |
| 11    | 中根 直哉 | C-2-1000m | 10   | <a href="#">削除</a> |
| 12    | 成瀬 真  | C-2-1000m | 21   | <a href="#">削除</a> |
| 19    | 緒方 健悟 | C-2-1000m | 3    | <a href="#">削除</a> |
| 21    | 橋本 和東 | C-2-1000m | 12   | <a href="#">削除</a> |
| 24    | 岡本 光  | JC-2-500m | 27   | <a href="#">削除</a> |

図 7 ペア登録画面。

## 4.7 登録内容確認画面

この画面では、今まで入力した内容の確認ができる。もし間違っているような内容があった場合、その画面に戻って修正を行うことができる。全ての内容を確認した場合、選手・エントリー登録は終了となる。

### 登録内容確認

大学名：京都大学

#### 選手名簿

| 大学内No | 選手名    | 種目 | 学年 |
|-------|--------|----|----|
| 1     | 高尾 充政  | C  | 4  |
| 2     | 田中 晴哉  | C  | 4  |
| 3     | 三津 海童  | C  | 4  |
| 4     | 池端 隆彦  | K  | 4  |
| 5     | 市川 公惇  | K  | 4  |
| 6     | 奥平 雅樹  | K  | 4  |
| 7     | 村田 翔   | K  | 4  |
| 8     | 大花 ちなみ | WK | 4  |

図 8 登録内容確認画面。

次は、管理者による操作である。

## 4.8 大学読みがな指定画面

この画面は選手名簿を作るための前準備をするためのページである。選手名簿は大学名順で並び替えを行うが、漢字ではソートを行うことができないため、ここで読みがなを指定する。「参加大学一覧」では、各大学が登録した *players* テーブルから、*DISTINCT* により重複を除いて参加大学一覧を引っ張ってきている。上の表に応じて、参加大学の読みがなを登録する。

### 参加大学一覧・読み仮名指定

[管理ページに戻る](#)

#### 参加大学一覧

| 大学名     |
|---------|
| 立命館大学   |
| 宮崎大学    |
| 同志社大学   |
| 神戸大学    |
| 福井工業大学  |
| 関西医科大学  |
| 流通科学大学  |
| 武庫川女子大学 |
| 鹿屋体育大学  |
| 京都大学    |
| 関西学院大学  |
| 甲南大学    |

#### 大学読み仮名一覧

| 大学名  | 読みがな     | 操作                                |
|------|----------|-----------------------------------|
| 京都大学 | きょうとだいがく | <input type="button" value="削除"/> |

図 9 大学読みがな指定画面.

## 4.9 選手名簿作成画面

この画面では、選手名簿を作成を行う。*players, bibs, reads* を自然結合してテーブルを表示し、大学読みがな順、学年順、種目順にソートされている。「ゼッケンを割り振る」を押下することによってこの表示順に *bibs* テーブルにゼッケン番号が挿入される。選手登録の際には、選手を大学名と大学内ナンバーによって区別していたが、大会中は選手をゼッケン番号によって区別する。そのため、ゼッケン番号と選手番号を結びつけるビューを作成しておけば良い。

### 現在の選手登録内容(選手名簿)

[管理ページに戻る](#)

ゼッケン番号を割り振る・再度割り振る

※初回割り振り時や、ゼッケン番号が抜けている時、順番がおかしい時に押してください。

| ゼッケンNo | 大学名    | 名前     | 種目 | 学年 |
|--------|--------|--------|----|----|
| 1      | 鹿屋体育大学 | 佐藤 博幸  | 3  | C  |
| 2      | 鹿屋体育大学 | 森黒 開   | 3  | K  |
| 3      | 鹿屋体育大学 | 田原 瞭太  | 3  | K  |
| 4      | 鹿屋体育大学 | 森 愛奈   | 3  | WK |
| 5      | 鹿屋体育大学 | 石川 義活  | 2  | C  |
| 6      | 鹿屋体育大学 | 宮原 直也  | 2  | C  |
| 7      | 鹿屋体育大学 | 下屋敷 泰成 | 2  | C  |
| 8      | 鹿屋体育大学 | 橋沼 新   | 2  | K  |
| 9      | 鹿屋体育大学 | 松村 圭祐  | 2  | K  |
| 10     | 鹿屋体育大学 | 林田 薫   | 2  | WK |
| 11     | 鹿屋体育大学 | 溝口 朋美  | 2  | WK |
| 12     | 鹿屋体育大学 | 鈴木 涼太  | 1  | C  |
| 13     | 鹿屋体育大学 | 岩男 凌   | 1  | C  |

図 10 選手名簿作成画面。

## 4.10 エントリー確認・レース一覧画面

この画面ではエントリーの確認とレースがいくつ作成されてるかを確認することができる。エントリー数は *COUNT* によって数えられ表示されている。「一覧」ボタンを押下することによってエントリー一覧を確認することができる。また、「一括自動作成する」ボタンを押下することによってエントリー数によって自動的にレースを作成することができる。例えば、参加艇数が 72 81 艇の場合には 1 次 *H9* 組, *H5* 組, *SF3* 組, *F1* 組となる。ここで *race* テーブルに自動挿入されるデータは、レースナンバーがマイナス、時刻が *null* となっているため、作成・更新ボタンを押下してレース *No* や時刻を指定する。

### エントリー一覧・レース作成

[管理ページに戻る](#)

一括自動作成する

※初回のみ実行してください。レースNoや時間がリセットされます。

| レース名       | エントリー艇(組)数 | エントリー | 組数  |   |    |   | 操作    |
|------------|------------|-------|-----|---|----|---|-------|
|            |            |       | 1次H | H | SF | F |       |
| K-1-1000m  | 81         | 一覧    | 9   | 5 | 3  | 1 | 作成・更新 |
| C-1-1000m  | 44         | 一覧    | 0   | 5 | 3  | 1 | 作成・更新 |
| WK-1-500m  | 43         | 一覧    | 0   | 5 | 3  | 1 | 作成・更新 |
| WC-1-500m  | 6          | 一覧    | 0   | 0 | 0  | 1 | 作成・更新 |
| JK-1-500m  | 12         | 一覧    | 0   | 2 | 0  | 1 | 作成・更新 |
| JC-1-500m  | 7          | 一覧    | 0   | 0 | 0  | 1 | 作成・更新 |
| JWK-1-500m | 6          | 一覧    | 0   | 0 | 0  | 1 | 作成・更新 |
| JWC-1-500m | 0          | 一覧    | 0   | 0 | 0  | 0 | 作成・更新 |
| K-2-1000m  | 36         | 一覧    | 0   | 4 | 3  | 1 | 作成・更新 |

図 11 エントリー確認・レース一覧画面.

## 4.11 レース編集画面

この画面では、レースごとの詳しいデータを変更・指定する。運営者は、会議などに行い、レースをどの順番で (レース *No*)、いつ (時刻) 行うかを決定する。そうして作成されたタイムテーブルに基づいてレース *No* や時刻を入力してテーブルのデータを変更していく。先ほどの自動作成でレースナンバーがマイナスとなっているものはレース *No* 「未定」と表示されているので、手動で正の値のレースナンバーを指定してレースを作成する。

### レース作成・更新

レース名：K-1-1000m

[エントリー一覧・レース作成に戻る](#)

### 現在のレース内容

| レースNo | 時刻    | レース名      | 組     | 操作   |
|-------|-------|-----------|-------|--|
| 1     | 9:35  | K-1-1000m | 1次H 1 | レースNoを <input type="text"/> 、時刻を <input type="text"/> に <input type="button" value="変更"/> レースを <input type="button" value="削除"/> |
| 2     | 9:40  | K-1-1000m | 1次H 2 | レースNoを <input type="text"/> 、時刻を <input type="text"/> に <input type="button" value="変更"/> レースを <input type="button" value="削除"/> |
| 3     | 9:50  | K-1-1000m | 1次H 3 | レースNoを <input type="text"/> 、時刻を <input type="text"/> に <input type="button" value="変更"/> レースを <input type="button" value="削除"/> |
| 4     | 10:00 | K-1-1000m | 1次H 4 | レースNoを <input type="text"/> 、時刻を <input type="text"/> に <input type="button" value="変更"/> レースを <input type="button" value="削除"/> |
| 5     | 10:10 | K-1-1000m | 1次H 5 | レースNoを <input type="text"/> 、時刻を <input type="text"/> に <input type="button" value="変更"/> レースを <input type="button" value="削除"/> |
| 6     | 10:20 | K-1-1000m | 1次H 6 | レースNoを <input type="text"/> 、時刻を <input type="text"/> に <input type="button" value="変更"/> レースを <input type="button" value="削除"/> |
| 7     | 10:30 | K-1-1000m | 1次H 7 | レースNoを <input type="text"/> 、時刻を <input type="text"/> に <input type="button" value="変更"/> レースを <input type="button" value="削除"/> |
| 8     | 10:40 | K-1-1000m | 1次H 8 | レースNoを <input type="text"/> 、時刻を <input type="text"/> に <input type="button" value="変更"/> レースを <input type="button" value="削除"/> |

図 12 レース編集画面.

## 4.12 抽選実施画面

この画面では、コンピュータによる抽選を行う。以下のアルゴリズムに基づいて抽選を行っている。

1. 大学内で選手の登録順をシャッフルする。
2. シャッフルされた順番をもとに、各大学の選手を均等になるように組に割り振る。
3. 組内でシャッフルを行う。
4. 同じ大学が隣り合っていた場合は再度シャッフルする。
5. ある程度 (100 回) 行っても隣り合っていた場合は諦める。

即席で作ったアルゴリズムなので、大分無駄がある。どうしても隣り合ってしまう場合などは判定してループの回数を減らすべきであろう。

こうして作成されたスタートリスト (*players, bibs, races, participates* の結合) は、大会期間に多くアクセスされるため、ビューを作成しておくが良い。

### 抽選実施・組み合わせ確認

[管理ページに戻る](#)

抽選を実施する。

※抽選を実施します。既に組み合わせがある場合は新たに抽選します。

### 現在の組み合わせ内容

| レースNo | レース名      | 組   |   | レーン | ゼッケン | 名前     | 大学     |
|-------|-----------|-----|---|-----|------|--------|--------|
| 1     | K-1-1000m | 1次H | 1 | 1   | 83   | 上中 健士朗 | 京都大学   |
| 1     | K-1-1000m | 1次H | 1 | 2   | 112  | 國師 大朗  | 神戸大学   |
| 1     | K-1-1000m | 1次H | 1 | 3   | 24   | 藤井 凌   | 関西学院大学 |
| 1     | K-1-1000m | 1次H | 1 | 4   | 128  | 塩谷 陸人  | 同志社大学  |
| 1     | K-1-1000m | 1次H | 1 | 5   | 64   | 宇佐美 阜太 | 関西学院大学 |
| 1     | K-1-1000m | 1次H | 1 | 6   | 159  | 中野 雄太  | 福井工業大学 |
| 1     | K-1-1000m | 1次H | 1 | 7   | 193  | 縄 空    | 立命館大学  |
| 1     | K-1-1000m | 1次H | 1 | 8   | 3    | 田原 瞭太  | 鹿屋体育大学 |
| 1     | K-1-1000m | 1次H | 1 | 9   | 69   | 溝口 昂平  | 関西学院大学 |
| 2     | K-1-1000m | 1次H | 2 | 1   | 189  | 白田 隆之  | 立命館大学  |
| 2     | K-1-1000m | 1次H | 2 | 2   | 34   | 小田 正明  | 関西学院大学 |
| 2     | K-1-1000m | 1次H | 2 | 3   | 164  | 佐藤 悦也  | 福井工業大学 |

図 13 抽選実施画面。

以下の画面は大会期間中に操作する。

### 4.13 順位入力画面

この画面では審判が順位を入力する。

**順位入力**

レースNo:

1着:  レーン

2着:  レーン

3着:  レーン

4着:  レーン

5着:  レーン

6着:  レーン

7着:  レーン

8着:  レーン

9着:  レーン

※既に結果がある場合は上書きされます。

[管理ページに戻る](#)

[トップページに戻る](#)

図 14 順位入力画面。

### 4.14 タイム入力画面

この画面では審判がタイムを入力する。

**タイム入力**

レースNo:

1着:

2着:

3着:

4着:

5着:

6着:

7着:

8着:

9着:

例: 4:07:19

※既に結果がある場合は上書きされます。

[管理ページに戻る](#)

[トップページに戻る](#)

図 15 タイム入力画面。



## 4.15 速報画面

この画面では、大会前には抽選によって作成されたスタートリストの閲覧をすることができ、結果入力後には結果速報を閲覧することができる。結果結果入力後はビューを作成しておいたほうが検索速度も速くなるだろう。

### スタートリスト・記録速報

レース名:  ステージ:  組:

※決勝は組「すべて」を選択してください。

### 検索結果

| レースNo | レース名      | ステージ | 組 | レーン | ゼッケン | 名前    | 大学     | 順位 | タイム         |
|-------|-----------|------|---|-----|------|-------|--------|----|-------------|
| 3     | K-1-1000m | 1次H  | 3 | 1   | 153  | 坂口 幸司 | 福井工業大学 | 3  | 00:04:02.45 |
| 3     | K-1-1000m | 1次H  | 3 | 2   | 185  | 名倉 光晟 | 立命館大学  | 2  | 00:04:02.33 |
| 3     | K-1-1000m | 1次H  | 3 | 3   | 67   | 野村 悠太 | 関西学院大学 | 5  | 00:04:05.14 |
| 3     | K-1-1000m | 1次H  | 3 | 4   | 108  | 渡瀬 亮太 | 神戸大学   | 4  | 00:04:04.11 |
| 3     | K-1-1000m | 1次H  | 3 | 5   | 23   | 籠谷 洋輔 | 関西学院大学 | 6  | 00:04:07.13 |
| 3     | K-1-1000m | 1次H  | 3 | 6   | 131  | 四戸 宗  | 同志社大学  | 8  | 00:04:12.34 |
| 3     | K-1-1000m | 1次H  | 3 | 7   | 15   | 安藤 久騎 | 鹿屋体育大学 | 7  | 00:04:10.34 |
| 3     | K-1-1000m | 1次H  | 3 | 8   | 47   | 稲井 勝紀 | 関西学院大学 | 1  | 00:04:01.11 |
| 3     | K-1-1000m | 1次H  | 3 | 9   | 75   | 奥平 雅樹 | 京都大学   | 9  | 00:04:15.23 |

[トップページに戻る](#)

図 16 速報画面。

## 5 工夫点

自分がシステムを使用する立場になって、できるだけ使いやすいようにインターフェースを調整した。エラー処理では、投げられる可能性のある例外をしっかりと拾ったり、*if* 文で *null* 処理を行ったりしてユーザーがエラーをわかりやすいようにした。*SQL* 文では、挿入整合性を考えて関係を設計し、結合を駆使してユーザーにわかりやすいようにデータを提示した。*COUNT* や *DISTINCT*、*ORDERBY*、*MAX* などの関数も用いている。索引については、1つの大会では多くのデータを保持しないため、索引は作っていないが、今後、過去の大会のデータを集めてデータベースを作成する際には索引を作成する必要があるだろう。ビューに関しては、スタートリストや結果など、速報ページにて多くアクセスされるものについて作成している。先述の通り、ペア登録においてトランザクション処理を行い、1人が複数のペアを組むことを防いでいる。全体的にセキュリティ面の処理がまだ甘い。*SQL* インジェクション対策やログインなど、今後実装していきたい。

## 6 感想

システム概要で述べたように、大会運営側という立場を利用してデータベースを扱いたいと考え、この実験を選択した。初めは過去の大会の結果のみをまとめたデータベースを作りたいと考えていたが、データベースを学ぶうち、*web* エントリーの段階からデータベースを利用できることに気づいた。必然的に扱う関係が多くなり、大会準備作業などで作成するページが多くなってしまったが、今回このように1通りのシステムを構築できたことは良い経験になったと思う。今後、このようなシステムを実際に運用していくかはまた理事会で検討するが、セキュリティ対策もちゃんと取り組み実用化したいと自分は考えている。*JavaServlet* をホームページ上で動かすには *VPN* サーバーなどを借りる必要が出てきてしまうため、*PHP* などを用いての実装も考えている。いずれにせよ、今回 *SQL* を用いてデータベースを扱うことができたのはいい経験となったと思う。今後も学習を続けてシステム実用化に向けて頑張りたい。