第1章　データベースのスキーマの設計

1.1　初期スキーマの作成

本レポートでは、ドラえもんに登場する人物の情報からデータベースを作成する。
登場人物の情報を表すために必要な属性を書き出して、以下の初期スキーマを作成した。

登場人物 (人物番号, 氏名, 服の色, 年齢, 世代, 身長)

1.1.1　属性の説明

人物番号 各登場人物に固有の人物号を表す。4 桁の数字による文字列となる。

氏名 各登場人物の氏名を表す。最大 12 文字の文字列となる。

服の色 各登場人物が主に着ている服の色を表す。最大 3 文字の文字列となる。R

年齢 各登場人物の年齢を表す。2 桁の数字による文字列となる。

世代 各登場人物が子どもか大人なのかを表す。最大３文字の文字列となる。

身長 各登場人物の身長を表す。３桁の数字による文字列となる。

1.2　リレーションに格納されるデータ

登場人物リレーションに格納されるデータは、以下の条件を満たす。

1. １．登場人物には固有の人物番号が割り当てられており、人物番号が同じである登場人物が複数存在することはない。
2. ２．氏名と身長の両方が同じ登場人物が複数存在することはない。
3. ３．各登場人物には固有の服の色が割り当てられており、一つの氏名に対してただ一色、服の色が決まるものとする。
4. ４．各登場人物の年齢によって、登場人物は、１つの世代に振り分けられるとする。

1.2.1　候補キー・主キー

条件 1 より、{ 人物番号 } は登場人物リレーションの候補キーとなる。
条件 2 より、{ 氏名, 身長 } も登場人物リレーションの候補キーとなる。
ここでは、２つの候補キーのうち、人物番号を主キーとする。
主キー属性に下線を引いた初期スキーマは、以下の通りである。

登場人物 (人物番号, 氏名, 服の色, 年齢, 世代, 身長)

1.2.2　関数従属性・多値従属性

条件３より、氏名 → 服の色 の関数従属性が存在する。
条件４より、年齢 → 世代 の関数従属性が存在する。

1.3　リレーションスキーマの正規化

登場人物リレーションが全ての正規形を満たすように、正規化を行う。

1.3.1　第 1 正規形

「登場人物リレーションは、全ての属性が単一の値を持つため、第 1 正規形を満たす。」

1.3.2　第 2 正規形

「登場人物リレーションは、候補キーの一部の属性から候補キー以外の属性への関数従属性が存在するため、第 2 正規形を満たさない。第 2 正規形を満たすようにするために、登場人物リレーションを、以下のように分解する。」

人物 (人物番号氏名, 年齢, 世代), 洋服 (氏名, 服の色)

1.3.3　第 3 正規形

「人物リレーションは、年齢 → 世代 の関数従属性により、候補キー以外の属性から候補キー以外の属性への関数従属性が存在するため、第 3 正規形を満たさない。第 3 正規形を満たすようにするために、人物リレーションを、以下のように分解する。」

個人 (人物番号, 氏名, 年齢, 世代, 身長)
年 (年齢, 世代　)

1.3.4　ボイス・コッド正規形

個人リレーション、洋服リレーション、年リレーションは、候補キー以外の属性から候補キーへの関数従属性が存在しないため、ボイス・コッド正規形を満たしている。

1.3.5　第 4 正規形

個人リレーション、年リレーション、洋服リレーションは、多値従属性が存在しないため第４正規形を満たしている。

1.3.6　第 5 正規形

個人リレーション、洋服リレーション、年リレーションは、結合従属性が存在しないため、第 5 正規形を満たしている。

1.4　正規化後のリレーションスキーマ

最終的に、以下のリレーションスキーマが得られた。

個人 (人物番号, 氏名, 年齢, 身長) 洋服 (氏名, 服の色) 年 (年齢, 世代)
最終的に得られたリレーションスキーマには、下記の参照整合制約（外部キー制約）が存在する。

1. 氏名
2. 年齢

第2章　データベースの作成

2.1　テーブルの定義

前章で設計した以下のリレーションスキーマにもとづいてデータベースを作成する。

個人 (人物番号, 氏名, 年齢, 身長)
洋服 (氏名, 服の色)
年 (年齢, 世代)

参照整合性制約（外部キー制約）

1. 個人の氏名（洋服の氏名を参照）
2. 個人の年齢（年の年齢を参照）

テーブル名・属性名をアルファベットに置き換えて、以下のテーブルを作成する。

parson(id, name, age, height)
clothes(name, color)
year(age, generation)

2.2　テーブルの作成

以下のコマンドを使用して、個人（person）テーブルを作成した。

ソースコード 2.1: 個人テーブルの作成のコマンド

```
1 create table person(
2    id varchar(4) not null unique,
3    name varchar(12) not null,
4    age int2,
5         height int3,
6    primary key( id )
7 );
```

以下のコマンドを使用して、洋服（clothes）テーブルを作成した。

ソースコード 2.2: 洋服テーブルの作成のコマンド

```
1 create table clothes(
2         name varchar(12) not null,
3         color varchar(3) not null,
4    primary key( name )
5 );
```

以下のコマンドを使用して、年齢（year）テーブルを作成した。

ソースコード 2.3: 年齢テーブルの作成コマンド

```
  create table year(
          age int2,
          generation varchar(2) not null,
          primary key( age )
);
```

以下のコマンドを使用して、個人（person）テーブルに参照整合性制約を追加した。

ソースコード 2.4: 個人テーブルへの参照整合性制約の設定のコマンド

```
alter table person add constraint person_clothes_key foreign key (name) references
    clothes (name);
```

以下のコマンドを使用して、個人（person）テーブルに参照整合性制約を追加した。

ソースコード 2.5: 個人テーブルへの参照整合性制約の設定コマンド

```
 alter table person add constraint person_year_key foreign key (age) references year (
     age);
```

以下のコマンドを使用して、全てのテーブルに対して、ウェブサーバへの利用権限を設定した。

ソースコード 2.6: 個人・洋服・年テーブルへの利用権限の設定のコマンド

```
grant all on person to apache;
grant all on clothes to apache;
grant all on year to apache;
```

2.3 初期データの挿入

以下のテキストファイル・コマンドを使用して、個人（person）テーブルに初期データを挿入した。

```
0001, ジャイ子, 8, 112
0002, スネ夫, 10, 135
0003, のび太, 10, 140
0004, セワシ, 10, 136
0005, しずか, 10, 138
0006, のびすけ, 41, 178
0007, 玉子, 38, 165
0008, 先生, 43, 179
0009, 出来杉, 10, 140
0010, ジャイアン, 10, 143
0011, スネ夫ママ, 37, 173
0012, のびすけ, 10, 138
0013, 神成さん, 68, 169
```

ソースコード 2.8: 個人テーブルへの初期データの挿入

```
insert into person values('0001','ジャイ子','8','112');
insert into person values('0002','スネ夫','10','135');
insert into person values('0003','のび太','10','140');
insert into person values('0004','セワシ','10','136');
insert into person values('0005','しずか','10','138');
insert into person values('0006','のびすけ','41','178');
insert into person values('0007','玉子','38','165');
insert into person values('0008','先生','43','179');
insert into person values('0009','出来杉','10','140');
insert into person values('0010','ジャイアン','10','143');
```

```
11 insert into person values('0011','スネ夫ママ','37','173');
12 insert into person values('0012','ノビスケ ','10','138');
13 insert into person values('0013','神成さん','68','169');
```

以下のテキストファイルとコマンドを使用して、洋服（clothes）テーブルに初期データを挿入した。

```
1  ジャイ子, 黄色
2  スネ夫, 水色
3  のび太,黄色
4  セワシ,橙色
5  しずか,ピンク
6  のびすけ,青
7  玉子,ピンク
8  先生,茶色
9  出来杉,水色
10 ジャイアン,橙色
11 スネ夫ママ,紫
12 ノビスケ,青
13 神成さん,茶色
```

ソースコード 2.10: 洋服テーブルへの初期データの挿入

```
1  insert into clothes values('ジャイ子', '黄色');
2  insert into clothes values('スネ夫', '水色');
3  insert into clothes values('のび太','黄色');
4  insert into clothes values('セワシ','橙色');
5  insert into clothes values('しずか','ピンク');
6  insert into clothes values('のびすけ','青');
7  insert into clothes values('玉子','ピンク');
8  insert into clothes values('先生','茶色');
9  insert into clothes values('出来杉','水色');
10 insert into clothes values('ジャイアン','橙色');
11 insert into clothes values('スネ夫ママ','紫');
12 insert into clothes values('ノビスケ','青');
13 insert into clothes values('神成さん','茶色');
```

以下のテキストファイルとコマンドを使用して、年齢（year）テーブルに初期データを挿入した。

```
1  8, 子ども
2  10, 子ども
3  10, 子ども
4  10, 子ども
5  10, 子ども
6  41, 大人
7  38, 大人
8  43, 大人
9  10, 子ども
10 10, 子ども
11 37, 大人
12 10, 子ども
13 68, 大人
```

ソースコード 2.12: テーブルへの初期データの挿入

```
1  insert into year values('8', '子ども');
2  insert into year values('10', '子ども');
3  insert into year values('10', '子ども');
4  insert into year values('10', '子ども');
5  insert into year values('10', '子ども');
6  insert into year values('41', '大人');
7  insert into year values('38', '大人');
8  insert into year values('43', '大人');
```

```
 9 insert into year values('10', '子ども');
10 insert into year values('10', '子ども');
11 insert into year values('37', '大人');
12 insert into year values('10', '子ども');
13 insert into year values('68', '大人');
```

2.4 テーブルの格納データの確認

SQL を使って個人（person）テーブルに格納されている全てのデータを表示すると、以下のようになる。

```
ayks2821=> select * from person;
 id | name | age | height
------+------------+-----+--------
0001 | ジャイ子 | 8 | 112
0002 | スネ夫 | 10 | 135
0003 | のび太 | 10 | 140
0004 | セワシ | 10 | 136
0005 | しずか | 10 | 138
0006 | のびすけ | 41 | 178
0007 | 玉子 | 38 | 165
0008 | 先生 | 43 | 179
0009 | 出来杉 | 10 | 140
0010 | ジャイアン | 10 | 143
0011 | スネ夫ママ | 37 | 173
0012 | のびすけ | 10 | 138
0013 | 神成さん | 68 | 169
(13 rows)
```

SQL を使って洋服（clothes）テーブルに格納されている全てのデータを表示すると、以下のようになる。

```
ayks2821=> select * from clothes;
 name | color
------------+--------
ジャイ子 | 黄色
スネ夫 | 水色
のび太 | 黄色
しずか | ピンク
のびすけ | 青
玉子 | ピンク
先生 | 茶色
出来杉 | 水色
ジャイアン | 橙色
スネ夫ママ | 紫
神成さん | 茶色
セワシ | 橙色
ノビスケ | 青
(13 rows)
```

SQL を使って年齢（year）テーブルに格納されている全てのデータを表示すると、以下のようになる。

```
ayks2821=> select * from year;
age | generation
-----+------------
 8 | 子ども
 10 | 子ども
 41 | 大人
 38 | 大人
 43 | 大人
 37 | 大人
 68 | 大人
(7 rows)
```

第3章　データベースへの問い合わせ

※ 作成したデータベースに対して、データベースを利用する際に使われる問い合わせの具体例を考えて、その問い合わせに対する SQL と実行結果を最低３つ示す。問い合わせの具体例は何でも構わないが、出力結果が想定した問い合わせと一致しているか（正しい結果が出力されているか）を確認すること。なるべく GROUP BY（+HAVING）や入れ子型問い合わせ（相関あり・なし）などの、授業で学習したやや複雑な構文を用いた SQL を含めることが望ましい。

2 章で作成したデータベースに対して、以下のような SQL を使った問い合わせのテストを行った。

3.1　問い合わせ 1

問い合わせ：
身長が 170cm 以上かつ大人である人物の人物番号と名前、身長、世代の一覧を出力する。

SQL：

```
select person.id, person.name, person.height, year.generation from person, year where person.height <=
```

実行結果：

```
ayks2821=> select person.id, person.name, person.height, year.generation from person, year where person
  id  |   name   | height | generation
------+----------+--------+------------
 0007 | 玉子     |    165 | 大人
 0013 | 神成さん |    169 | 大人
(2 rows)
```

3.2　問い合わせ 2

問い合わせ：
同じ年齢の人物の身長の平均と年齢、同じ年齢の人数の一覧を出力する。

SQL：

```
select age, AVG(height), count(*) from person group by age having count(*) >= 2;
```

実行結果：

```
ayks2821=> select age, AVG(height), count(*) from person group by age having count(*) >= 2;
 age |         avg          | count
-----+----------------------+-------
  10 | 122.6250000000000000 |     8
(1 row)
```

3.3 問い合わせ 3

問い合わせ：
小学生（6 歳から 12 歳）の間で、平均の身長より身長が高い人物の人物番号と名前と身長を昇順に出力する。

SQL：

```
select id, name, height from person where height > (select AVG(height) from person where age between 6
```

実行結果：

```
ayks2821=> select id, name, height from person where height > (select AVG(height) from person where age
  id  |    name    | height
------+------------+--------
 0002 | スネ夫     |    135
 0004 | セワシ     |    136
 0005 | しずか     |    138
 0012 | ノビスケ   |    139
 0003 | のび太     |    140
 0009 | 出来杉     |    140
 0010 | ジャイアン |    143
(7 rows)
```

第4章　Webインターフェースの作成

4.1　作成したインターフェースのメニューページ

2 章で作成したデータベースを利用するためのインターフェースを開発した。
作成したインターフェースのメニューページの URL は、下記の通りである。

`http://db.tom.ai.kyutech.ac.jp/~ayks2821/reportmenu.html`

4.2　作成したインターフェースの構成

個人と洋服と年齢を結合して一覧表示する機能を作成した。
個人テーブルへのデータの挿入・削除・更新のための機能を作成した。
洋服テーブルと年齢テーブルについては、更新の頻度が少ないため、データの挿入・削除・更新の機能は作成していない。
また、人物の情報を名前（名前の一部でも可）から検索できる機能を作成した。
インターフェースを構成するファイルの役割や階層構造は、下記の通りである。

- メニューページ（reportmenu.html）
  - 個人・洋服・年齢の一覧表示（person_list.php）
  - 個人のデータ追加の入力フォーム（person_add_form.php）
    - 個人のデータ追加の処理実行（person_add.php）
  - 個人のデータ削除の選択（person_delete_form.php）
    - 個人のデータ削除の処理実行（person_delete.php）
  - 個人のデータ更新の選択（person_update_form1.php）
    - 個人のデータ更新の入力フォーム（person_update_form2.php）
      - 個人のデータ更新の処理実行（person_update.php）
  - 個人の検索の入力フォーム（person_search_form.php）
    - 個人の検索結果の表示（person_search_result.php）

4.3　メニューページ

メニューページは、上記の階層構造においてメニューページの下位のページである、個人・洋服・年齢の一覧表示（person_list.php）、個人のデータ追加（person_add_form.php）、個人のデータ削除（person_delete_form.php）、個人のデータ更新（person_update_form1.php）、個人の検索（person_search_form.php）の各ページへのリンクを含む。

ソースコード 4.1: menu.html

```
1  <HTML>
2  <HEAD>
3    <TITLE>データ操作メニュー</TITLE>
4    <META http-equiv="Content-Type" content="text/html; charset=UTF-8">
5  </HEAD>
6  <BODY>
7
8  操作メニュー<BR>
9
10 <UL>
11     <LI><A HREF="employee_list.php">従業員の一覧表示</A>
12     <LI><A HREF="employee_add_form.php">従業員のデータ追加</A>
13     <LI><A HREF="employee_delete_form.php">従業員のデータ削除</A>
14     <LI><A HREF="employee_update_form1.php">従業員のデータ更新</A>
15     <LI><A HREF="employee_search_form.php">従業員の検索（部門名での検索）</A>
16 </UL>
17
18 </BODY>
19 </HTML>
```

図 4.1 は、ウェブブラウザでメニューページを表示したときのスクリーンショットを示したものである。

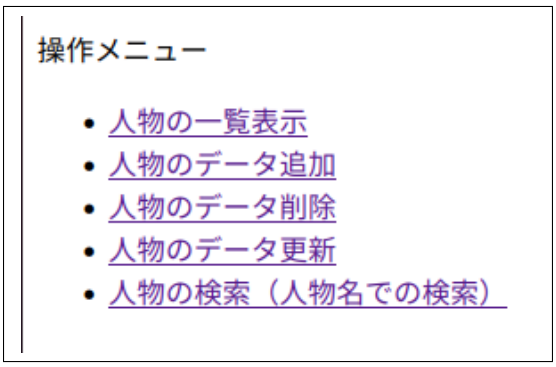


図 4.1: メニューページの表示結果

4.4 個人の一覧表示

4.4.1 個人の一覧表示（person_list.php）

全人物の一覧を表示する PHP プログラムを含むページである。個人と洋服と年齢のテーブルを結合して、人物番号にもとづいて昇順で並べて表示する。また、全体で何件の人物のデータが存在しているかの情報を、人物一覧の後に表示する。

プログラムの 27 行目で人物の一覧を作成する SQL 文を作成して、30 行目でその SQL 文を実行している。

ソースコード 4.2: 人物の一覧表示のための SQL

```
1  select person.id, person.name, person.age, person.height, clothes.color, year.
       generation from person, clothes, year where person.name = clothes.name and person.
       age = year.age order by id asc";
```

この問い合わせは、前ページでの利用者の入力によって変化するようなことはなく、毎回同じ問い合わせを実行するため、ソースファイル中で固定の SQL 文を文字列として設定している。

ソースコード 4.3: person_list.php

```
1  <HTML>
2  <HEAD>
3    <TITLE>人物リスト</TITLE>
4    <META http-equiv="Content-Type" content="text/html; charset=UTF-8">
5  </HEAD>
6  <BODY>
7
8  <CENTER>
9
10 検索結果を表示します。<BR><BR>
11
12 <!-- ここからPHPのスクリプト始まり -->
13 <?php
14
15 // データベースに接続
16 // ※ your_db_name のところは自分のデータベース名に書き換える
17 $conn = pg_connect( "dbname=ayks2821" );
18
19 // 接続が成功したかどうか確認
20 if ( $conn == null )
21 {
22    print( "データベース接続処理でエラーが発生しました。<BR>" );
23    exit;
24 }
25
26 // SQLを作成
27 $sql = "select person.id, person.name, person.age, person.height, clothes.color, year.
       generation from person, clothes, year where person.name = clothes.name and person.
       age = year.age order by id asc";
28
29 // Queryを実行して検索結果をresultに格納
30 $result = pg_query( $conn, $sql );
31 if ( $result == null )
32 {
33    print( "クエリー実行処理でエラーが発生しました。<BR>" );
34    exit;
35 }
36
37 // 検索結果の行数・列数を取得
38 $rows = pg_num_rows( $result );
39 $cols = pg_num_fields( $result );
40
41
42 // 検索結果をテーブルとして表示
43 print( "<TABLE BORDER=1>\n" );
44
45 // 各列の名前を表示
46 print( "<TR>" );
47 print( "<TH>人物番号</TH>" );
48 print( "<TH>名前</TH>" );
49 print( "<TH>年齢</TH>" );
50 print( "<TH>身長</TH>" );
51 print( "<TH>服の色</TH>" );
52 print( "<TH>世代</TH>" );
53 print( "</TR>\n" );
54
55 // 各行のデータを表示
56 for ( $j=0; $j<$rows; $j++ )
57 {
58    print( "<TR>" );
59    for ( $i=0; $i<$cols; $i++ )
60    {
```

```
61          // j行i列のデータを取得
62          $data = pg_fetch_result( $result, $j, $i );
63
64          // セルに列の名前を表示
65          print( "<TD> $data </TD>" );
66      }
67      print( "</TR>\n" );
68  }
69
70  // ここまででテーブル終了
71  print( "</TABLE>" );
72  print( "<BR>\n" );
73
74
75  // 検索件数を表示
76  print( "以上、$rows 件のデータを表示しました。<BR>\n" );
77
78
79  // 検索結果の開放
80  pg_free_result( $result );
81
82  // データベースへの接続を解除
83  pg_close( $conn );
84
85  ?>
86  <!-- ここまででPHPのスクリプト終わり -->
87
88  <BR>
89  <A HREF="reportmenu.html">操作メニューに戻る</A>
90
91  </CENTER>
92
93  </BODY>
94  </HTML>
```

4.4.2 人物の一覧表示の実行例

図 4.2 は、一覧表示の操作を行ったときのスクリーンショットを示したものである。

検索結果を表示します。

人物番号	名前	年齢	身長	服の色	世代
0001	ジャイ子	8	112	黄色	子ども
0002	スネ夫	10	135	水色	子ども
0003	のび太	10	140	黄色	子ども
0004	セワシ	10	136	橙色	子ども
0005	しずか	10	138	ピンク	子ども
0006	のびすけ	41	178	青	大人
0007	玉子	38	165	ピンク	大人
0008	先生	43	179	茶色	大人
0009	出来杉	10	140	水色	子ども
0010	ジャイアン	10	143	橙色	子ども
0011	スネ夫ママ	37	173	紫	大人
0012	ノビスケ	10	139	青	子ども
0013	神成さん	68	169	茶色	大人

以上、13 件のデータを表示しました。

操作メニューに戻る

図 4.2: 個人の一覧表示の実行結果

4.5 個人の追加

4.5.1 個人のデータ追加の入力フォーム（person_add_form.php）

個人の追加を行う PHP プログラムを含むページである。指定された人物番号、名前、年齢、身長のインスタンスを人物テーブルに追加する。名前、年齢、身長をフォームに入力することによって指定する。人物番号はインスタンス内の最大値から自動的に最大値の次の番号に指定されるようになっている。

ソースコード 4.4: person_add_form.php

```
1  <HTML>
2  <HEAD>
3    <TITLE>人物データ追加フォーム（動的生成版）</TITLE>
4    <META http-equiv="Content-Type" content="text/html; charset=UTF-8">
5  </HEAD>
6  <BODY>
7
8  人物データ 追加フォーム<BR><BR>
9
10 <FORM ACTION="person_add.php" METHOD="GET">
11
12 <!-- ここからPHPのスクリプト始まり -->
13 <?php
14
15 // データベースに接続
16 // ※ your_db_name のところは自分のデータベース名に書き換える
17 $conn = pg_connect( "dbname=ayks2821" );
18
19 // 接続が成功したかどうか確認
20 if ( $conn == null )
21 {
```

```
22      print( "データベース接続処理でエラーが発生しました。<BR>" );
23      exit;
24  }
25
26
27  // 最も大きな従業員番号を取り出すSQLの作成
28  $sql = "select max(id) from person";
29
30  // Queryを実行して検索結果をresultに格納
31  $result = pg_query( $conn, $sql );
32  if ( $result == null )
33  {
34      print( "クエリー実行処理bでエラーが発生しました。<BR>" );
35      exit;
36  }
37
38  // 最大の従業員番号を取得
39  if ( pg_num_rows( $result ) > 0 )
40      $max_id = pg_fetch_result( $result, 0, 0 );
41  $max_id ++;
42
43  // 従業員番号の初期値を指定して入力エリアを作成
44  print( "人物番号：\n" );
45  printf( "<INPUT TYPE=text SIZE=4 NAME=id VALUE=%04s>", $max_id ); // 必ず4桁で出力、空
        白があれば0で埋める
46  print( "<BR>\n" );
47
48  // 検索結果の開放
49  pg_free_result( $result );
50
51
52  // 部門一覧を取得するSQLの作成
53  $sql = "select person.id, clothes.color, person.name, person.age, person.height from
        person, clothes where person.name = clothes.name";
54
55  // Queryを実行して検索結果をresultに格納
56  $result = pg_query( $conn, $sql );
57  if ( $result == null )
58  {
59      print( "クエリー実行処理cでエラーが発生しました。<BR>" );
60      exit;
61  }
62
63  // 検索結果の行数を取得
64  $rows = pg_num_rows( $result );
65
66  // 部門の数だけ選択肢を出力
67  print( "服の色：\n" );
68  for ( $i=0; $i<$rows; $i++ )
69  {
70      $name= pg_fetch_result( $result, $i, 0 );
71      $color = pg_fetch_result( $result, $i, 1 );
72      printf( "<INPUT TYPE=\"radio\" NAME=\"name\" VALUE=\"%s\"> %s </INPUT>\n", $name,
            $color );
73  }
74
75  // 検索結果の開放
76  pg_free_result( $result );
77
78  // データベースへの接続を解除
79  pg_close( $conn );
80
81  ?>
82  <!-- ここまででPHPのスクリプト終わり -->
```

```
83
84  <BR>
85
86  名前：
87  <INPUT TYPE="text" SIZE="24" NAME="name">
88
89  年齢：
90  <INPUT TYPE="text" SIZE="4" NAME="age">
91
92  身長：
93  <INPUT TYPE="text" SIZE="4" NAME="height">
94
95  <BR><BR>
96  <INPUT TYPE="submit" VALUE="送信"><BR>
97
98  </FORM>
99
100 </BODY>
101 </HTML>
```

4.5.2 個人のデータ追加の処理実行（person_add.php）

前のページのフォームに対して入力された、以下の情報を受け取る。プログラムの 12〜15 行目で、これらのデータを取得して、各変数に代入する。

- 人物番号（id） → $id
- 名前（name） → $name
- 年齢（age） → $age
- 身長（height） → $height

プログラムの 30 行目で、人物テーブルにインスタンスを追加するための SQL 文を作成する。SQL 文の %s, %s, %s, %s には、変数 $id, $name, $age, $height の値が挿入される。30 行目で、作成した SQL 文を実行する。

ソースコード 4.5: 人物の追加のための SQL

```
1  insert into person values( %1, %2, %3, %4 )
```

ソースコード 4.6: person_add.php

```
1  <HTML>
2  <HEAD>
3    <TITLE>人物データ追加処理スクリプト</TITLE>
4    <META http-equiv="Content-Type" content="text/html; charset=UTF-8">
5  </HEAD>
6  <BODY>
7
8  <!-- ここからPHPのスクリプト始まり -->
9  <?php
10
11 // フォームから渡された引数を取得
12 $id = $_GET[ id ];
13 $name = $_GET[ name ];
14 $age = $_GET[ age ];
15 $height = $_GET[ height ];
16
```

```
17
18 // データベースに接続
19 // ※ your_db_name のところは自分のデータベース名に書き換える
20 $conn = pg_connect( "dbname=ayks2821" );
21
22 // 接続が成功したかどうか確認
23 if ( $conn == null )
24 {
25     print( "データベース接続処理でエラーが発生しました。<BR>" );
26     exit;
27 }
28
29 // データ挿入のSQLを作成
30 $sql = sprintf( "insert into person( id, name, age, height ) values( '%s', '%s', '%s',
       '%s');", $id, $name, $age, $height);
31
32 // 確認用のメッセージ表示
33 print( "クエリー「" );
34 print( $sql );
35 print( "」を実行します。<BR>" );
36
37 // Queryを実行して検索結果をresultに格納
38 $result = pg_query( $conn, $sql );
39 if ( $result == null )
40 {
41     print( "クエリー実行処理aでエラーが発生しました。 <BR>" );
42     exit;
43 }
44
45 // 検索結果の開放
46 pg_free_result( $result );
47
48 // データベースへの接続を解除
49 pg_close( $conn );
50
51 ?>
52 <!-- ここまででPHPのスクリプト終わり -->
53
54 データの追加処理が完了しました。<BR>
55 <BR>
56 <A HREF="reportmenu.html">操作メニューに戻る</A>
57
58 </BODY>
59 </HTML>
```

4.5.3 人物の追加の実行例

図 4.3 は、追加処理前のスクリーンショットを示したものである。

図 4.3: 追加処理前の実行結果

図 4.4 は、追加処理完了後のスクリーンショットを示したものである。

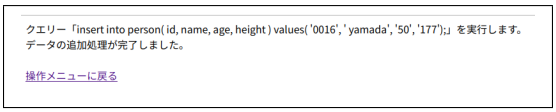


図 4.4: 追加処理後の実行結果

4.6 人物の削除

4.6.1 人物のデータ削除の選択（person_delete_form.php）

人物の削除を行う PHP プログラムを含むページである。人物番号のラジオボタンの選択することで人物のインスタンスをリレーションから削除する。

ソースコード 4.7: person_delete_form.php

```
1  <HTML>
2  <HEAD>
3    <TITLE>人物の削除フォーム（動的生成版）</TITLE>
4    <META http-equiv="Content-Type" content="text/html; charset=UTF-8">
5  </HEAD>
6  <BODY>
7
8  <CENTER>
9
10 人物データ 削除フォーム<BR><BR>
11
12 削除したい人物を選択して送信ボタンを押してください。<BR><BR>
13
14 <FORM ACTION="person_delete.php" METHOD="GET">
15
16 <!-- ここからPHPのスクリプト始まり -->
17 <?php
18
19 // データベースに接続
20 // ※ your_db_name のところは自分のデータベース名に書き換える
21 $conn = pg_connect( "dbname=ayks2821" );
22
23 // 接続が成功したかどうか確認
24 if ( $conn == null )
25 {
26     print( "データベース接続処理でエラーが発生しました。<BR>" );
27     exit;
28 }
29
30 // SQLを作成
31 $sql = "select person.id, person.name, person.age, person.height, clothes.color, year.
       generation from person, clothes, year where person.name=clothes.name and person.age
       =year.age order by id";
32
33 // Queryを実行して検索結果をresultに格納
34 $result = pg_query( $conn, $sql );
35 if ( $result == null )
36 {
37     print( "クエリー実行処理でエラーが発生しました。<BR>" );
38     exit;
39 }
40
41 // 検索結果の行数・列数を取得
42 $rows = pg_num_rows( $result );
43 $cols = pg_num_fields( $result );
```

```
44 |
45 |
46 | // 検索結果をテーブルとして表示
47 | print( "<TABLE BORDER=1>\n" );
48 |
49 | // 各列の名前を表示
50 | print( "<TR>" );
51 | print( "<TH>人物番号</TH>" );
52 | print( "<TH>名前</TH>" );
53 | print( "<TH>年齢</TH>" );
54 | print( "<TH>身長</TH>" );
55 | print( "<TH>服の色</TH>" );
56 | print( "<TH>世代</TH>" );
57 | print( "</TR>\n" );
58 |
59 | // 各行のデータを表示
60 | for ( $j=0; $j<$rows; $j++ )
61 | {
62 |     print( "<TR>" );
63 |
64 |     // 従業員番号と選択のためのラジオボタンを表示
65 |     $data = pg_fetch_result( $result, $j, 0 );
66 |     print( "<TD> <INPUT TYPE=\"radio\" NAME=\"id\" VALUE=\"$data\"> $data </INPUT> </TD
       >\n" );
67 |
68 |     // 残りの属性値を表示（$iが1から始まっている点に注意）
69 |     for ( $i=1; $i<$cols; $i++ )
70 |     {
71 |         // j行i列のデータを取得
72 |         $data = pg_fetch_result( $result, $j, $i );
73 |
74 |         // セルに列の名前を表示
75 |         print( "<TD> $data </TD>" );
76 |     }
77 |
78 |     print( "</TR>\n" );
79 | }
80 |
81 | // ここまででテーブル終了
82 | print( "</TABLE>" );
83 | print( "<BR>\n" );
84 |
85 |
86 | // 検索件数を表示
87 | print( "以上、$rows 人の人物が登録されています。<BR>\n" );
88 |
89 |
90 | // 検索結果の開放
91 | pg_free_result( $result );
92 |
93 | // データベースへの接続を解除
94 | pg_close( $conn );
95 |
96 | ?>
97 | <!-- ここまででPHPのスクリプト終わり -->
98 |
99 | <BR>
100 | <INPUT TYPE="submit" VALUE="送信"><BR>
101 |
102 | </FORM>
103 |
104 | <BR>
105 | <A HREF="reportmenu.html">操作メニューに戻る</A>
106 |
```

```
107 </CENTER>
108
109 </BODY>
110 </HTML>
```

4.6.2 人物のデータ削除の処理実行（employee_delete.php）

前のページのフォームに対して入力された、以下の情報を受け取る。プログラムの 12 行目で、これらのデータを取得して、各変数に代入する。

- 人物番号（id） → $id;

ソースコード 4.8: 入力した人物番号の情報を削除するための SQL

```
1 $sql = "delete from person where id = '".$id."'";
```

ソースコード 4.9: person_delete.php

```
1  <HTML>
2  <HEAD>
3    <TITLE>従業員データ削除処理スクリプト</TITLE>
4    <META http-equiv="Content-Type" content="text/html; charset=UTF-8">
5  </HEAD>
6  <BODY>
7
8  <!-- ここからPHPのスクリプト始まり -->
9  <?php
10
11 // フォームから渡された引数を取得
12 $id = (string)$_GET[ id ];
13
14 // データベースに接続
15 // ※ your_db_name のところは自分のデータベース名に書き換える
16 $conn = pg_connect( "dbname=ayks2821" );
17
18 // 接続が成功したかどうか確認
19 if ( $conn == null )
20 {
21     print( "データベース接続処理でエラーが発生しました。<BR>" );
22     exit;
23 }
24
25 // データ削除のSQLを作成
26 // ※ 課題：どのようなSQLを作成したら良いか自分で考えてみよ
27 $sql = "delete from person where id = '".$id."'";
28
29 // 確認用のメッセージ表示
30 print( "クエリー「" );
31 print( $sql );
32 print( "」を実行します。<BR>" );
33
34 // Queryを実行して検索結果をresultに記録
35 $result = pg_query( $conn, $sql );
36 if ( $result == null )
37 {
38     print( "クエリー実行処理でエラーが発生しました。<BR>" );
39     exit;
40 }
41
42 // 検索結果の開放
```

```
43 pg_free_result( $result );
44
45 // データベースへの接続を解除
46 pg_close( $conn );
47
48 ?>
49 <!-- ここまででPHPのスクリプト終わり -->
50
51 データの削除処理が完了しました。<BR>
52 <BR>
53 <A HREF="reportmenu.html">操作メニューに戻る</A>
54
55 </BODY>
56 </HTML>
```

4.6.3 人物の削除の実行例

図 4.5 は、削除処理前のスクリーンショットを示したものである。

図 4.5: 削除処理前の実行結果

図 4.6 は、削除処理完了後のスクリーンショットを示したものである。

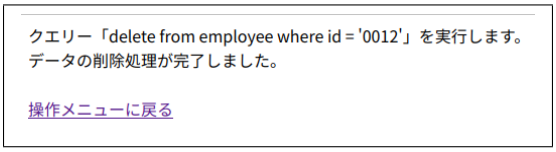


図 4.6: 削除処理完了後の実行結果

4.7 人物の更新

4.7.1 人物のデータ更新の選択（person_update_form1.php）

人物の一覧を作成し、人物番号のラジオボタンで更新したい人物を指定するページである。

ソースコード 4.10: person_update_form1.php

```
<HTML>
<HEAD>
  <TITLE>人物の更新フォーム（動的生成版）</TITLE>
  <META http-equiv="Content-Type" content="text/html; charset=UTF-8">
</HEAD>
<BODY>

<CENTER>

人物データ 更新フォーム<BR><BR>

更新したい人物を選択して送信ボタンを押してください。<BR><BR>

<FORM ACTION="person_update_form2.php" METHOD="GET">

<!-- ここからPHPのスクリプト始まり -->
<?php

// データベースに接続
// ※ your_db_name のところは自分のデータベース名に書き換える
$conn = pg_connect( "dbname=ayks2821" );

// 接続が成功したかどうか確認
if ( $conn == null )
{
    print( "データベース接続処理でエラーが発生しました。<BR>" );
    exit;
}

// SQLを作成
$sql = "select person.id, person.name, person.age, person.height, clothes.color, year.
    generation from person, clothes, year where person.name = clothes.name and person.
    age = year.age order by id";

// Queryを実行して検索結果をresultに格納
$result = pg_query( $conn, $sql );
if ( $result == null )
{
    print( "クエリー実行処理でエラーが発生しました。<BR>" );
    exit;
}

// 検索結果の行数・列数を取得
$rows = pg_num_rows( $result );
$cols = pg_num_fields( $result );


// 検索結果をテーブルとして表示
print( "<TABLE BORDER=1>\n" );

// 各列の名前を表示
print( "<TR>" );
print( "<TH>人物番号</TH>" );
print( "<TH>名前</TH>" );
print( "<TH>年齢</TH>" );
print( "<TH>身長</TH>" );
```

```
55 print( "<TH>服の色</TH>" );
56 print( "<TH>世代</TH>" );
57 print( "</TR>\n" );
58
59 // 各行のデータを表示
60 for ( $j=0; $j<$rows; $j++ )
61 {
62     print( "<TR>" );
63
64     // 従業員番号と選択のためのラジオボタンを表示
65     $data = pg_fetch_result( $result, $j, 0 );
66     print( "<TD> <INPUT TYPE=\"radio\" NAME=\"id\" VALUE=\"$data\"> $data </INPUT> </TD
           >\n" );
67
68     // 残りの属性値を表示（$iが1から始まっている点に注意）
69     for ( $i=1; $i<$cols; $i++ )
70     {
71         // j行i列のデータを取得
72         $data = pg_fetch_result( $result, $j, $i );
73
74         // セルに列の名前を表示
75         print( "<TD> $data </TD>" );
76     }
77
78     print( "</TR>\n" );
79 }
80
81 // ここまででテーブル終了
82 print( "</TABLE>" );
83 print( "<BR>\n" );
84
85
86 // 検索件数を表示
87 print( "以上、$rows 人の人物が登録されています。<BR>\n" );
88
89
90 // 検索結果の開放
91 pg_free_result( $result );
92
93 // データベースへの接続を解除
94 pg_close( $conn );
95
96 ?>
97 <!-- ここまででPHPのスクリプト終わり -->
98
99 <BR>
100 <INPUT TYPE="submit" VALUE="送信"><BR>
101
102 </FORM>
103
104 <BR>
105 <A HREF="reportmenu.html">操作メニューに戻る</A>
106
107 </CENTER>
108
109 </BODY>
110 </HTML>
```

4.7.2 人物のデータ更新の入力フォーム（person_update_form2.php）

ソースコード 4.11: person_update_form2.php

```
<HTML>
<HEAD>
  <TITLE>人物データ更新フォーム</TITLE>
  <META http-equiv="Content-Type" content="text/html; charset=UTF-8">
</HEAD>
<BODY>

人物データ 更新フォーム<BR><BR>

<FORM ACTION="person_update.php" METHOD="GET">

<!-- ここからPHPのスクリプト始まり -->
<?php

// 引数の人物番号を取得
$id = (string) $_GET[ id ];

// データベースに接続
// ※ your_db_name のところは自分のデータベース名に書き換える
$conn = pg_connect( "dbname=ayks2821" );

// 接続が成功したかどうか確認
if ( $conn == null )
{
   print( "データベース接続処理aでエラーが発生しました。<BR>" );
   exit;
}

// 指定された従業員番号の従業員情報を取得するSQLを作成
$sql = sprintf( "select id, name, age, height from person where id='%s'", $id );

// Queryを実行して検索結果をresultに記録
$result = pg_query( $conn, $sql );
if ( $result == null )
{
   print( "クエリー実行処理bでエラーが発生しました。<BR>" );
   exit;
}

// 従業員が見つからなければエラーメッセージを表示
if ( pg_num_rows( $result ) == 0 )
{
   print( "指定された人物番号のデータが見つかりません。<BR>\n" );
   exit;
}

// 検索結果の従業員の情報を変数に記録
$curr_id = pg_fetch_result( $result, 0, 0 );
$curr_name = pg_fetch_result( $result, 0, 1 );
$curr_age = pg_fetch_result( $result, 0, 2 );
$curr_height = pg_fetch_result( $result, 0, 3 );

// 検索結果の開放
pg_free_result( $result );

// 従業員番号を更新スクリプトに渡す
printf( "<INPUT TYPE=hidden NAME=id VALUE=%s>\n", $id );


// 部門一覧を取得するSQLの作成
$sql = "select name, color from clothes";

```

```
63 // Queryを実行して検索結果をresultに記録
64 $result = pg_query( $conn, $sql );
65 if ( $result == null )
66 {
67     print( "クエリー実行処理cでエラーが発生しました。<BR>" );
68     exit;
69 }
70
71 // 検索結果の行数を取得
72 $rows = pg_num_rows( $result );
73
74 // 部門の数だけ選択肢を出力
75
76 // 検索結果の開放
77 pg_free_result( $result );
78
79 // データベースへの接続を解除
80 pg_close( $conn );
81
82 // 番号の入力フィールドを出力
83 print( "<BR>\n" );
84 print( "人物番号：\n" );
85 printf( "<INPUT TYPE=text SIZE=24 NAME=id VALUE=\"%s\">\n", $curr_id );
86 print( "  \n" );
87
88 // 名前の入力フィールドを出力
89 print( "名前：\n" );
90 printf( "<INPUT TYPE=text SIZE=4 NAME=name VALUE=%s>\n", $curr_name );
91
92 // 年齢の入力フィールドを出力
93 print( "年齢：\n" );
94 printf( "<INPUT TYPE=text SIZE=4 NAME=age VALUE=%s>\n", $curr_age );
95
96 // 身長の入力フィールドを出力
97 print( "身長：\n" );
98 printf( "<INPUT TYPE=text SIZE=4 NAME=height VALUE=%s>\n", $curr_height );
99
100 ?>
101 <!-- ここまででPHPのスクリプト終わり -->
102
103 <BR>
104
105
106
107 <BR><BR>
108 <INPUT TYPE="submit" VALUE="送信"><BR>
109
110 </FORM>
111
112 </BODY>
113 </HTML>
```

4.7.3 人物のデータ更新の処理実行（person_update.php）

前のページのフォームに対して入力された、以下の情報を受け取る。プログラムの 11〜15 行目で、これらのデータを取得して、各変数に代入する。

- 人物番号（id） → $id
- 名前（name） → $name

- 年齢（age） → $age
- 身長（height） → $height

ソースコード 4.12: 入力した人物番号の情報を変更するための SQL

```
1 $sql = "update person set id='".$id."', name='".$name."', age='".$age."', height='".
      $height."' where person.id='".$id."'";
```

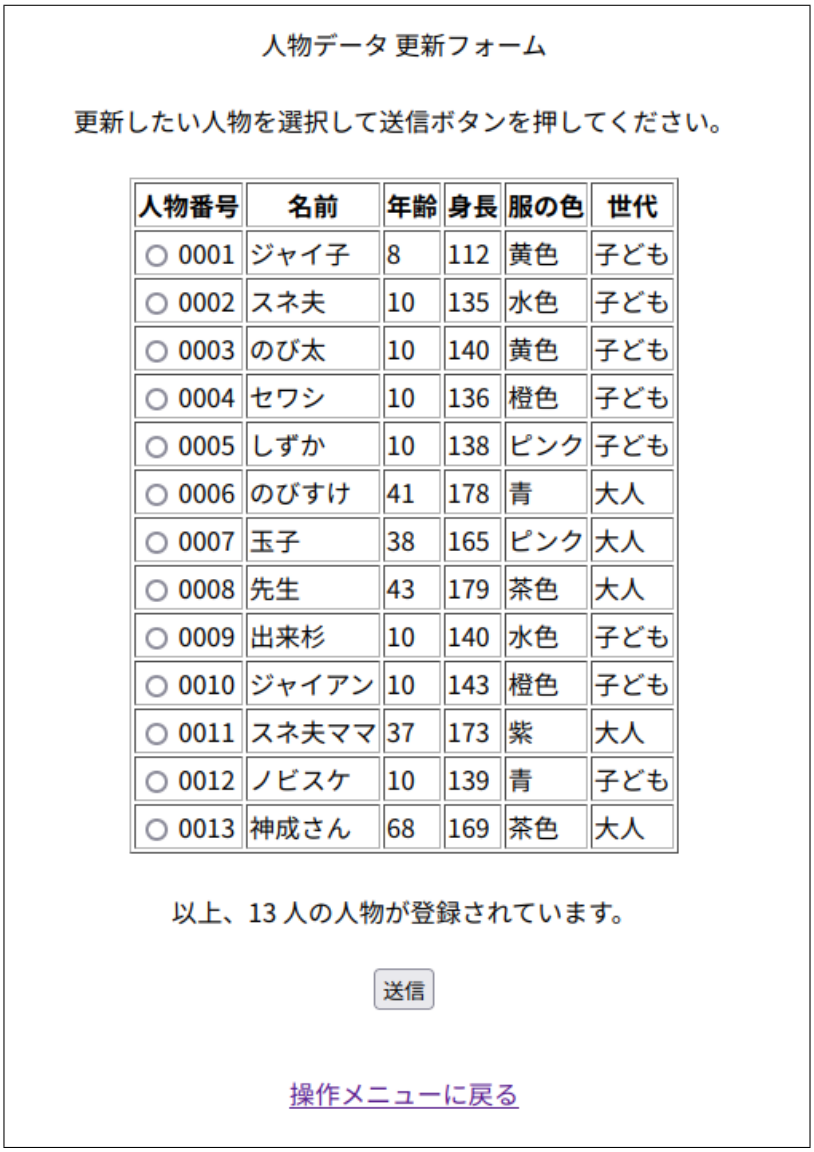
ソースコード 4.13: person_update.php

```
1  <HTML>
2  <HEAD>
3    <TITLE>人物データ更新処理スクリプト</TITLE>
4    <META http-equiv="Content-Type" content="text/html; charset=UTF-8">
5  </HEAD>
6  <BODY>
7
8  <!-- ここからPHPのスクリプト始まり -->
9  <?php
10
11 // フォームから渡された引数を取得
12 $id = $_GET[ id ];
13 $name = $_GET[ name ];
14 $age = $_GET[ age ];
15 $height = $_GET[ height ];
16
17 // データベースに接続
18 // ※ your_db_name のところは自分のデータベース名に書き換える
19 $conn = pg_connect( "dbname=ayks2821" );
20
21 // 接続が成功したかどうか確認
22 if ( $conn == null )
23 {
24     print( "データベース接続処理bでエラーが発生しました。<BR>" );
25     exit;
26 }
27
28 // データ更新のSQLを作成
29 // ※ 課題：どのようなSQLを作成したら良いか自分で考えてみよ
30 $sql = "update person set id='".$id."', name='".$name."', age='".$age."', height='".
       $height."' where person.id='".$id."'";
31
32 // 確認用のメッセージ表示
33 print( "クエリー「" );
34 print( $sql );
35 print( "」を実行します。<BR>" );
36
37 // Queryを実行して検索結果をresultに格納
38 $result = pg_query( $conn, $sql );
39 if ( $result == null )
40 {
41     print( "クエリー実行処理aでエラーが発生しました。<BR>" );
42     exit;
43 }
44
45 // 検索結果の開放
46 pg_free_result( $result );
47
48 // データベースへの接続を解除
49 pg_close( $conn );
50
51 ?>
```

```
52 <!-- ここまででPHPのスクリプト終わり -->
53
54 データの更新処理が完了しました。<BR>
55 <BR>
56 <A HREF="reportmenu.html">操作メニューに戻る</A>
57 </BODY>
58 </HTML>
```

4.7.4 人物のデータ更新の実行例

図 4.7〜4.11 は、人物「ノビスケ」の身長を「139」から「138」に更新する操作を行ったときの、一連のスクリーンショットを示したものである。



人物データ 更新フォーム

更新したい人物を選択して送信ボタンを押してください。

人物番号	名前	年齢	身長	服の色	世代
○ 0001	ジャイ子	8	112	黄色	子ども
○ 0002	スネ夫	10	135	水色	子ども
○ 0003	のび太	10	140	黄色	子ども
○ 0004	セワシ	10	136	橙色	子ども
○ 0005	しずか	10	138	ピンク	子ども
○ 0006	のびすけ	41	178	青	大人
○ 0007	玉子	38	165	ピンク	大人
○ 0008	先生	43	179	茶色	大人
○ 0009	出来杉	10	140	水色	子ども
○ 0010	ジャイアン	10	143	橙色	子ども
○ 0011	スネ夫ママ	37	173	紫	大人
○ 0012	ノビスケ	10	139	青	子ども
○ 0013	神成さん	68	169	茶色	大人

以上、13 人の人物が登録されています。

送信

操作メニューに戻る

図 4.7: 人物の更新の実行結果 (1)：選択フォームから更新する人物を選択

図 4.8: 人物の更新の実行結果 (2)：入力フォームに移り、選択した人物の現在の情報が表示される

図 4.9: 人物の更新の実行結果 (3)：入力フォームで、人物の情報を変更して、送信ボタンを押す

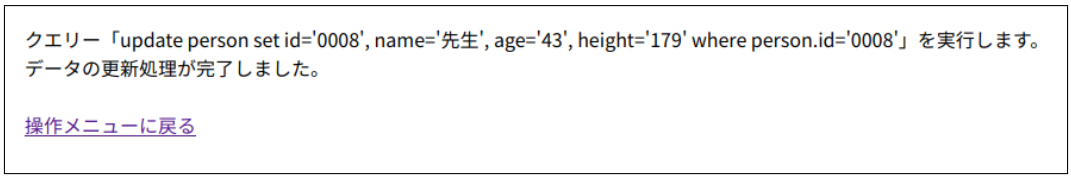


図 4.10: 人物の更新の実行結果 (4)：更新処理の実行結果が表示される

検索結果を表示します。

人物番号	名前	年齢	身長	服の色	世代
0001	ジャイ子	8	112	黄色	子ども
0002	スネ夫	10	135	水色	子ども
0003	のび太	10	140	黄色	子ども
0004	セワシ	10	136	橙色	子ども
0005	しずか	10	138	ピンク	子ども
0006	のびすけ	41	178	青	大人
0007	玉子	38	165	ピンク	大人
0008	先生	43	179	茶色	大人
0009	出来杉	10	140	水色	子ども
0010	ジャイアン	10	143	橙色	子ども
0011	スネ夫ママ	37	173	紫	大人
0012	ノビスケ	10	139	青	子ども
0013	神成さん	68	169	茶色	大人

以上、13 件のデータを表示しました。

操作メニューに戻る

図 4.11: 人物の更新の実行結果 (5)：一覧表示を行い、人物の情報が更新されていることを確認する

4.8 人物の検索

4.8.1 人物の検索の入力フォーム（person_search_form.php）

人物の検索を行う PHP プログラムを含むページである。名前（名前の一部でも可）を入力する。

4.8.2 人物の検索結果の表示（person_search_result.php）

前のページのフォームに対して入力された、以下の情報を受け取る。プログラムの 12 行目で、データを取得して、変数に代入する。

- 名前（name） → $name

プログラムの 26〜27 行目で、人物の検索を行うための SQL 文を作成する。変数 $id に文字列「ALL」が格納されている場合は、全ての部門の人物を表示するための SQL を実行する。

ソースコード 4.14: 全ての人物を表示するための SQL

```
1 $sql ="select * from person";
```

変数 $id に「ALL」以外の文字列が格納されている場合は、人物番号が $id に等しい人物のみを表示するための SQL を実行する。SQL 文の $id には、その変数の値が挿入される。

ソースコード 4.15: 入力された文字を検索するための SQL

```
1 select id, name, age, height from person where name like '%".$name."%';
```

ソースコード 4.16: person_search.php

```
1  <HTML>
2  <HEAD>
3    <TITLE>人物の検索結果</TITLE>
4    <META http-equiv="Content-Type" content="text/html; charset=UTF-8">
5  </HEAD>
6  <BODY>
7
8  <CENTER>
9
10 検索結果を表示します。<BR><BR>
11
12 <!-- ここからPHPのスクリプト始まり -->
13 <?php
14
15 // 検索フォームから渡された引数を取得
16 $id = (string)$_GET[ id ];
17
18 // データベースに接続
19 // ※ your_db_name のところは自分のデータベース名に書き換える
20 $conn = pg_connect( "dbname=ayks2821" );
21
22 // 接続が成功したかどうか確認
23 if ( $conn == null )
24 {
25     print( "データベース接続処理でエラーが発生しました。<BR>" );
26     exit;
27 }
28
29 //
       SQLを作成（検索フォームで全ての部門が指定された場合は全部門の従業員を検索し、それ以外の場合は、
```

```
30 if ( $id == "ALL" )
31    $sql ="select * from person"; // ※ 全部門の従業員を検索
32 else
33    $sql ="select * from person where id='$id'"; // ※ 指定された部門番号（
          $dept_no）の従業員を検索
34
35 // Queryを実行して検索結果をresultに格納
36 $result = pg_query( $conn, $sql );
37 if ( $result == null )
38 {
39    print( "クエリー実行処理でエラーが発生しました。<BR>" );
40    print( "クエリー「" . $sql . "」を実行。<BR>" );
41    exit;
42 }
43
44 // 検索結果の行数・列数を取得
45 $rows = pg_num_rows( $result );
46 $cols = pg_num_fields( $result );
47
48
49 // 検索結果をテーブルとして表示
50 print( "<TABLE BORDER=1>\n" );
51
52 // 各列の名前を表示
53 print( "<TR>" );
54 print( "<TH>人物番号</TH>" );
55 print( "<TH>名前</TH>" );
56 print( "<TH>年齢</TH>" );
57 print( "<TH>身長</TH>" );
58 print( "</TR>\n" );
59
60 // 各行のデータを表示
61 for ( $j=0; $j<$rows; $j++ )
62 {
63    print( "<TR>" );
64    for ( $i=0; $i<$cols; $i++ )
65    {
66       // j行i列のデータを取得
67       $data = pg_fetch_result( $result, $j, $i );
68
69       // セルに列の名前を表示
70       print( "<TD> $data </TD>" );
71    }
72    print( "</TR>\n" );
73 }
74
75 // ここまででテーブル終了
76 print( "</TABLE>" );
77 print( "<BR>\n" );
78
79
80 // 検索件数を表示
81 print( "以上、$rows 件のデータを表示しました。<BR>\n" );
82
83
84 // 検索結果の開放
85 pg_free_result( $result );
86
87 // データベースへの接続を解除
88 pg_close( $conn );
89
90 ?>
91 <!-- ここまででPHPのスクリプト終わり -->
92
```

```
93 <BR>
94 <A HREF=”reportmenu.html”>操作メニューに戻る</A>
95
96 </CENTER>
97
98 </BODY>
99 </HTML>
```

4.8.3 人物のデータ検索の実行例

図 4.12〜4.8 は、人物名「の」を入力し、検索を行ったときの、一連のスクリーンショットを示したものである。

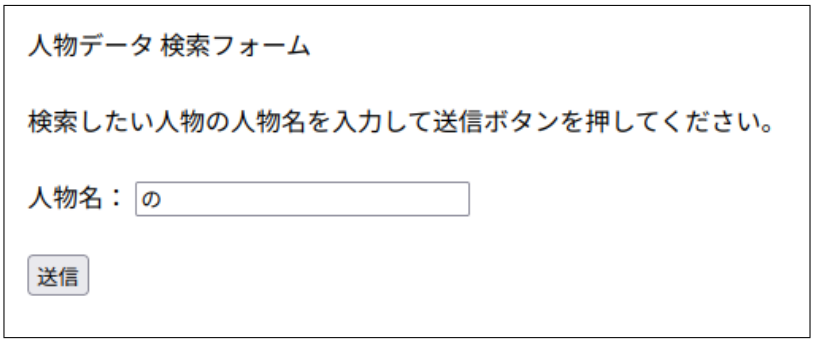


図 4.12: 検索する人物名を入力する画面の実行結果

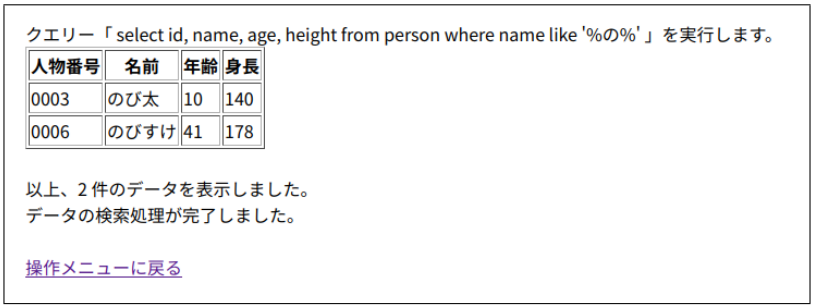


図 4.13: 検索操作完了後の実行結果

第5章　まとめ

今回のデータベースレポートでは、人物の情報を格納するデータベースを作成した。難しかった点を上げると、更新のプログラムを作成する時に、2 つのリレーションを 1 つのリレーションとして捉えて SQL で update することができなかった。改善策として、１つのリレーションだけを用いて、更新の操作を行ったが、そうすることで、ラジオボタンを使用することができなかった。操作することのできる Web ページを作成することができ、授業内容も含め、深く理解することにつながるレポートになった。