# データベース　課題3レポート


氏名　山久保孝亮
所属　大阪大学基礎工学部情報科学科ソフトウェア科学コース
メールアドレス　u327468b@ecs.osaka-u.ac.jp
学籍番号　09B22084

提出日　2025 年 1 月 8 日

# 1　課題 1

## 1.1　課題内容

5 人以下の学生が履修している科目の科目名とその科目を履修している学生数を求めよ

## 1.2　SQL の問い合わせ文

課題 1 の結果を返すための SQL の問い合わせ文は以下の図 1 のようになる.

```
WITH s_num(code,num)
  AS(
    SELECT code,COUNT(*)
    FROM registration
    GROUP BY code
    )
SELECT course.title,x.num
FROM s_num as x
JOIN course ON course.code = x.code
WHERE x.num <= 5;
```

図 1: 課題 1 の SQL の問い合わせ文

## 1.3　SQL 文の解法

課題 1 では最終的に科目名と受講生の人数を出力する必要がある.
科目名に関しては course の title を参照すればよいが，受講生の人数を表すカラムが存在しないので，GROUP BY を使用し各 code ごとにグループ表を作成する．その個数を COUNT(*) を使って num として計算する．これにより，各 code の表の行数を数えることができ，それはその code の受講人数を表す.
上記の処理を WITH を用いて行うが，処理後に作成された表を s_num とした.
最終的に出力するのは s_num のカラムの num と course のカラムの title であるが，これらを連結する必要がある．したがって，それぞれの表に共通するカラムである code を JOIN で結合する.
また，課題内容より，表示するのは受講生が 5 人以下の科目のみであるため WHERE において num が 5 以下という条件をつけた.

## 1.4 SQL 文の問い合わせ結果

問い合わせの結果は以下の図 2 のようになる．

**Query Output**

```
         title          | num
------------------------+-----
 Thesis Work            |   5
 Data Science           |   5
 Knowledge Engineering  |   5
 Computer Science Basics |   2
 Bio-Engineering Intro  |   2
(5 rows)
```

図 2: 課題 1 の問い合わせ結果

## 1.5 mongoGB の問い合わせ文

課題 1 の結果を返すための mongoGB の問い合わせ文は以下の図 3 のようになる．

```
db.rg.aggregate(
  {$group:{_id: '$code',count:{$sum: 1}}},
  {$match:{count:{$lte:5}}},
  {$lookup:{from:'cs',localField:'_id',foreignField:'code',as:'cs_num'}},
  {$project:{_id:0,count:1,cs_num:{title:1}}}
)
```

図 3: 課題 1 の mongoGB の問い合わせ文

## 1.6 mongoGB の解法

課題 1 の mongoGB の問い合わせ文は，1.2 の SQL 文から作成した．rg に対して aggregate を行った．各ステージの処理の詳細を以下で述べる．

1. $group ステージでは，1.2 の SQL における WITH 内の処理を行っている．具体的には，code ごとにグループ表を作り，count にはそれぞれの表の行の数だけ 1 を足す．これにより，count が各 code ごとの行数，即ち各講義の受講生の人数を表す．また，code は_id という名称になっている．

2. $match ステージでは，1.2 の SQL における WHERE の処理を行っている．具体的には，1 で求めた count が 5 以下のもののみを選択している．

3. $lookup ステージでは，1.2 の SQL における JOIN の処理を行っている．具体的には，1,2 で求めた受講生が 5 人以下の講義のみのグループ表と cs を結合している．cs とグループ表では，共通のフィールドは code と_id であるため，それらによって結合を行っている．結合後の表は cs_num という名称にした．

4. $project ステージでは，表示するフィールドの内容を指定している．_id は明示的に 0 にしないと表示されるため 0 としている．これは，以下の課題でも同様である．また，課題内容より，表示するものは科目名と受講生の人数なので 3 で求めた cs_num のフィールドである title と num を 1 とした．

### 1.7 mongoGB の問い合わせの結果

問い合わせの結果は以下の図 4 のようになる.

```
coursedb> ... ... ... ... ... [
  { count: 5, cs_num: [ { title: 'Knowledge Engineering' } ] },
  { count: 2, cs_num: [ { title: 'Bio-Engineering Intro' } ] },
  { count: 5, cs_num: [ { title: 'Thesis Work' } ] },
  { count: 5, cs_num: [ { title: 'Data Science' } ] },
  { count: 2, cs_num: [ { title: 'Computer Science Basics' } ] }
]
```

図 4: 課題 1 の mongoGB の問い合わせ結果

## 2 課題 2

### 2.1 課題内容

教室が空値である科目の科目名とその科目を履修している学生の学生名を求めよ

### 2.2 SQL の問い合わせ文

課題 2 の問い合わせに答えるための SQL 文は以下のとおりである.

```
SELECT course.title,student.name
FROM registration as rg
JOIN course ON course.code = rg.code
JOIN student ON student.number = rg.number
WHERE course.room IS null;
```

図 5: 課題 2 の SQL の問い合わせ文

### 2.3 SQL 文の解法

課題 2 では最終的に科目名と生徒名を出力する必要がある.
科目名は course に，生徒名は student テーブルに存在する．課題内容より，その科目を履修している学生名を表示しなければならないので，registration テーブルと前述の二つのテーブルを JOIN を用いて結合する．また，課題内容より，教室が空値である科目を選ぶ必要があるので WHERE では course テーブルの room が null であるという条件をつけた.

## 2.4 SQL 文の問い合わせ結果

問い合わせの結果は以下の図 2 のようになる．

**Query Output**

```
    title    |   name
-------------+----------
 Thesis Work | Aomori
 Thesis Work | Akita
 Thesis Work | Iwate
 Thesis Work | Yamagata
 Thesis Work | Miyagi
(5 rows)
```

図 6: 課題 1 の問い合わせ結果

## 2.5 mongoGB の問い合わせ文

課題 2 の結果を返すための mongoGB の問い合わせ文は以下のとおりである．

```
db.rg.aggregate(
  {$lookup:{from:'cs',localField:'code',foreignField:'code',as:'cs_rg'}},
  {$lookup:{from:'st',localField:'number',foreignField:'number',as:'st_rg'}},
  {$match:{'cs_rg.room':null}},
  {$project:{_id:0,cs_rg:{title:1},st_rg:{name:1}}}
)
```

図 7: 課題 2 の mongoGB の問い合わせ文

## 2.6 mongoGB の解法

課題 2 の mongoGB の問い合わせ文も，2.2 の SQL 文から作成した．rg に対して aggregate を行った．各ステージの処理の詳細を以下で述べる．

1. 一つ目の$lookup ステージでは，2.2 の SQL における JOIN の処理を行っている．共通するフィールドである code を用いて cs と rg を結合し，cs_rg という表を作成した．

2. 二つ目の$lookup ステージでは，2.2 の SQL における JOIN の処理を行っている．共通するフィールドである code を用いて cs と st を結合し，st_rg という表を作成した．

3. $match ステージでは，2.2 の SQL における WHERE の処理を行っている．room フィールドが null であるものを選択している．

4. $project ステージでは，表示するフィールドの内容を指定している．課題内容より，科目名を表す title と生徒名を表す name を 1 としている．

### 2.7 mongoGB の問い合わせの結果

問い合わせの結果は以下の図 2 のようになる.

```
coursedb> ... ... ... ... ... [
  {
    cs_rg: [ { title: 'Thesis Work' } ],
    st_rg: [ { name: 'Aomori' } ]
  },
  { cs_rg: [ { title: 'Thesis Work' } ], st_rg: [ { name: 'Akita' } ] },
  { cs_rg: [ { title: 'Thesis Work' } ], st_rg: [ { name: 'Iwate' } ] },
  {
    cs_rg: [ { title: 'Thesis Work' } ],
    st_rg: [ { name: 'Yamagata' } ]
  },
  {
    cs_rg: [ { title: 'Thesis Work' } ],
    st_rg: [ { name: 'Miyagi' } ]
  }
]
```

図 8: 課題 2 の問い合わせ結果

## 3 課題 3

### 3.1 課題内容

必修の科目について科目ごとに科目名と成績の最高点およびその成績を取った学生名を求めよ

### 3.2 SQL の問い合わせ文

課題 3 の問い合わせに答えるための SQL 文は以下のとおりである.

```
WITH max_grade(code,grade)
  AS (
    SELECT code,MAX(grade)
    FROM registration
    GROUP BY code
    )
SELECT course.title,max_grade.grade,student.name
FROM registration as rg
JOIN course ON course.code = rg.code
JOIN student ON student.number = rg.number
JOIN max_grade ON max_grade.code = rg.code
WHERE course.type = 'R' AND max_grade.grade = rg.grade;
```

図 9: 課題 3 の SQL の問い合わせ文

### 3.3 SQL 文の解法

課題 3 では最終的に科目名と最高点，学生名を出力する必要がある．
科目名は course テーブルの title，学生名は student テーブルの name を参照すればよいが，各科目の最高点を表すカラムを含んだテーブルは存在しない．したがって，WITH 内でこの最高点を計算する処理を行う．WITH により作成される表を max_grade とし，カラムは code と grade とした．GROUP BY を使って course テーブルの code ごとにグループ表を作成し，course テーブルの grade の最大値を MAX() を使って grade として選択する．これにより，このグループ表の grade は各科目ごとの成績の最高点を表すことができるようになる．
課題内容より，course テーブル，student テーブル，max_grade テーブルを registration テーブルと JOIN を用いて結合する．これにより，必要な情報を含んだ表を作成することができた．ただし，この表は registration テーブルのカラム grade と，max_grade テーブルのカラム grade を情報として持っている．
最後に WHERE の処理を記述する．まず必修の科目を選択するために type が'R' と一致するものを選択する．この段階で上記の表は必修科目のみに絞り込むことができた．次に，各科目の成績の最大値を表す max_grade テーブルのカラム grade と registration テーブルのカラム grade が一致している物を選択する．これにより，それぞれの grade が一致している物だけが，即ち registration テーブルの grade が最大値のものだけが選択される．

### 3.4 SQL 文の問い合わせ結果

問い合わせの結果は以下の図のようになる．

**Query Output**

```
        title         | grade |   name
----------------------+-------+----------
 Programming          |    99 | Miyagi
 Computer Architecture |    99 | Yamagata
 Database             |    95 | Akita
 Thesis Work          |    95 | Miyagi
(4 rows)
```

図 10: 課題 3 の問い合わせ結果

### 3.5 mongoGB の問い合わせ文

課題 3 の結果を返すための mongoGB の問い合わせ文は以下のとおりである．

**Your Command**

```
db.rg.aggregate([
  {$group:{_id:'$code',max_grade:{$max:'$grade'}}},
  {$lookup:{from:'rg',localField:'max_grade',foreignField:'grade',as:'m_rg'}},
  {$lookup:{from:'cs',localField:'_id',foreignField:'code',as:'rg_cs'}},
  {$lookup:{from:'st',localField:'m_rg.number',foreignField:'number',as:'rg_st'}},
  {$match:{'rg_cs.type':'R'}},

  {$project:{_id:0,max_grade:1,'rg_cs.title':1,'rg_st.name':1}}
])
```

図 11: 課題 3 の mongoGB の問い合わせ文

### 3.6 mongoGB の解法

課題 3 の mongoGB の問い合わせ文も，3.2 の SQL 文から作成した．rg に対して aggregate を行った．各ステージの処理の詳細を以下で述べる．

1. $group ステージでは，3.2 の SQL 文における WITH の処理を行っている．具体的には，code ごとにグループ表を作り，max_grade には grade の最大値を格納する．また，code は_id という名称になっている．

2. 一つ目の$lookup ステージでは，3.2 の SQL における JOIN の処理を行っている．ここでは，1 で作成したグループ表と rg を結合している．ここで rg と結合する理由としては，cs と st とそれぞれ結合する際，共通するフィールドが存在しないためである．今回は grade と max_grade を共通するフィールドとして結合している．また，結合後の表は m_rg とした．共通するフィールドとして max_grade を選んだ理由としては，各科目番号の成績の最大値のデータのみに絞り込むことができるためである．即ち，ここでは JOIN の処理だけでなく WHERE の AND の右側の処理も行っている．

3. 二つ目の$lookup ステージでも，3.2 の SQL における JOIN の処理を行っている．code と_id を用いて m_rg と cs を結合し，rg_cs という表を作成した．

4. 三つ目の$lookup ステージでも，3.2 の SQL における JOIN の処理を行っている．code と_id を用いて m_rg と st を結合し，rg_st という表を作成した．

5. $match ステージでは，3.2 の SQL における WHERE の処理を行っている．type フィールドが'R' であるものを選択している．これにより，必修の科目のみを選択することができる．

6. $project ステージでは，表示するフィールドの内容を指定している．課題内容より，成績の最大値を表す max_grade，科目名を表す title，生徒名を表す name を 1 としている．

### 3.7 mongoGB の問い合わせの結果

問い合わせの結果は以下の図 3 のようになる．

```
coursedb> ... ... ... ... ... ... ... ... [
  {
    max_grade: 99,
    rg_cs: [ { title: 'Programming' } ],
    rg_st: [ { name: 'Miyagi' } ]
  },
  {
    max_grade: 98,
    rg_cs: [ { title: 'Computer Architecture' } ],
    rg_st: [ { name: 'Yamagata' } ]
  },
  {
    max_grade: 97,
    rg_cs: [ { title: 'Database' } ],
    rg_st: [ { name: 'America' } ]
  },
  {
    max_grade: 96,
    rg_cs: [ { title: 'Thesis Work' } ],
    rg_st: [ { name: 'Aomori' } ]
  }
]
```

図 12: 課題 3 の問い合わせ結果

# 4　感想

今回の課題を通して難しく感じたことは，SQL 文と MongoGB の結合の違いについてである．SQL 文では 3 つ以上のテーブルを結合することができたが，MongoGB では二つのテーブルを結合することしかできない．したがって，結合するフィールドで工夫をする必要があった点が難しく感じた．