知識処理
第五回

知識の表現

その4

データベース（後編）

今日の内容

1. データベースの複数のテーブルを用いた問い合わせ
   - 他のテーブルの参照
   - サブクエリ
2. テーブルの結合
   - 内部結合（inner join）と
   - 外部結合（outer join）

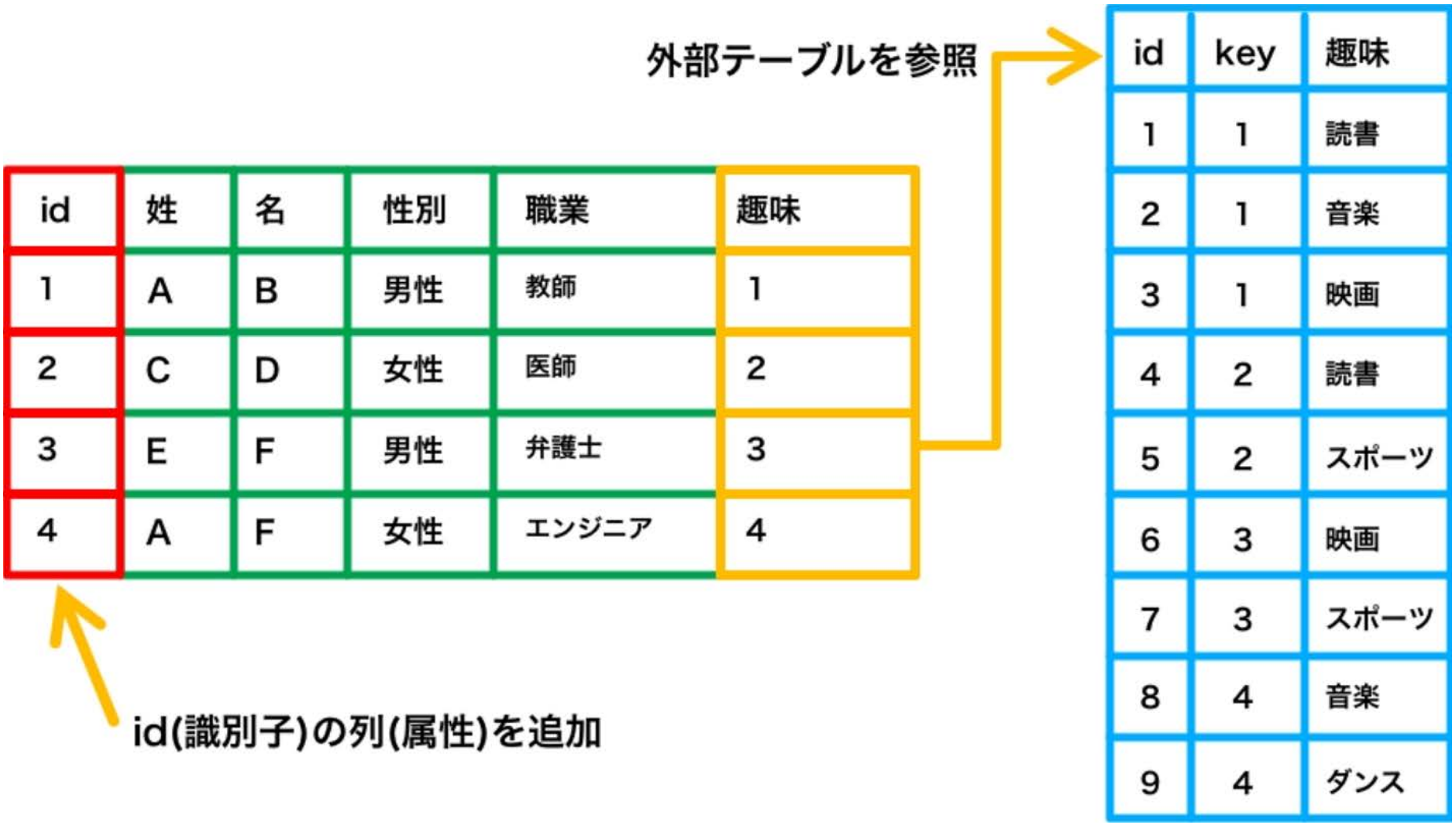
前回の問題

- 下のテーブル「友人帳」を正規化したテーブルを書いてください。
- 必要に応じて外部テーブルを作成してください。

姓	名	性別	職業	趣味
A	B	男性	教師	読書、音楽、映画
C	D	女性	医師	読書、スポーツ
E	F	男性	弁護士	映画、スポーツ
A	F	女性	エンジニア	音楽、ダンス

解答

- 以下のように、正規化されたテーブルを作る

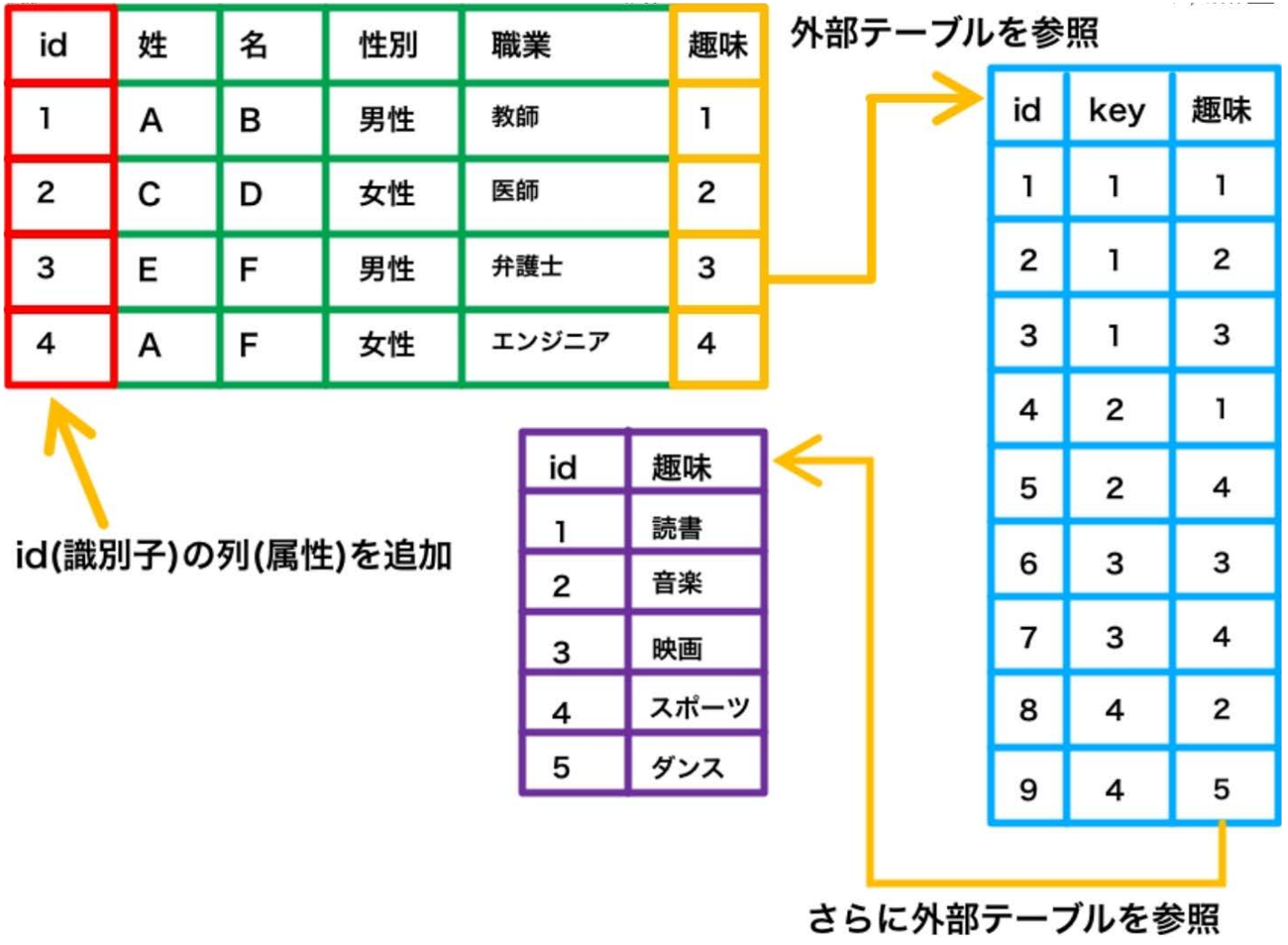


id	姓	名	性別	職業	趣味
1	A	B	男性	教師	1
2	C	D	女性	医師	2
3	E	F	男性	弁護士	3
4	A	F	女性	エンジニア	4

id	key	趣味
1	1	読書
2	1	音楽
3	1	映画
4	2	読書
5	2	スポーツ
6	3	映画
7	3	スポーツ
8	4	音楽
9	4	ダンス

別解

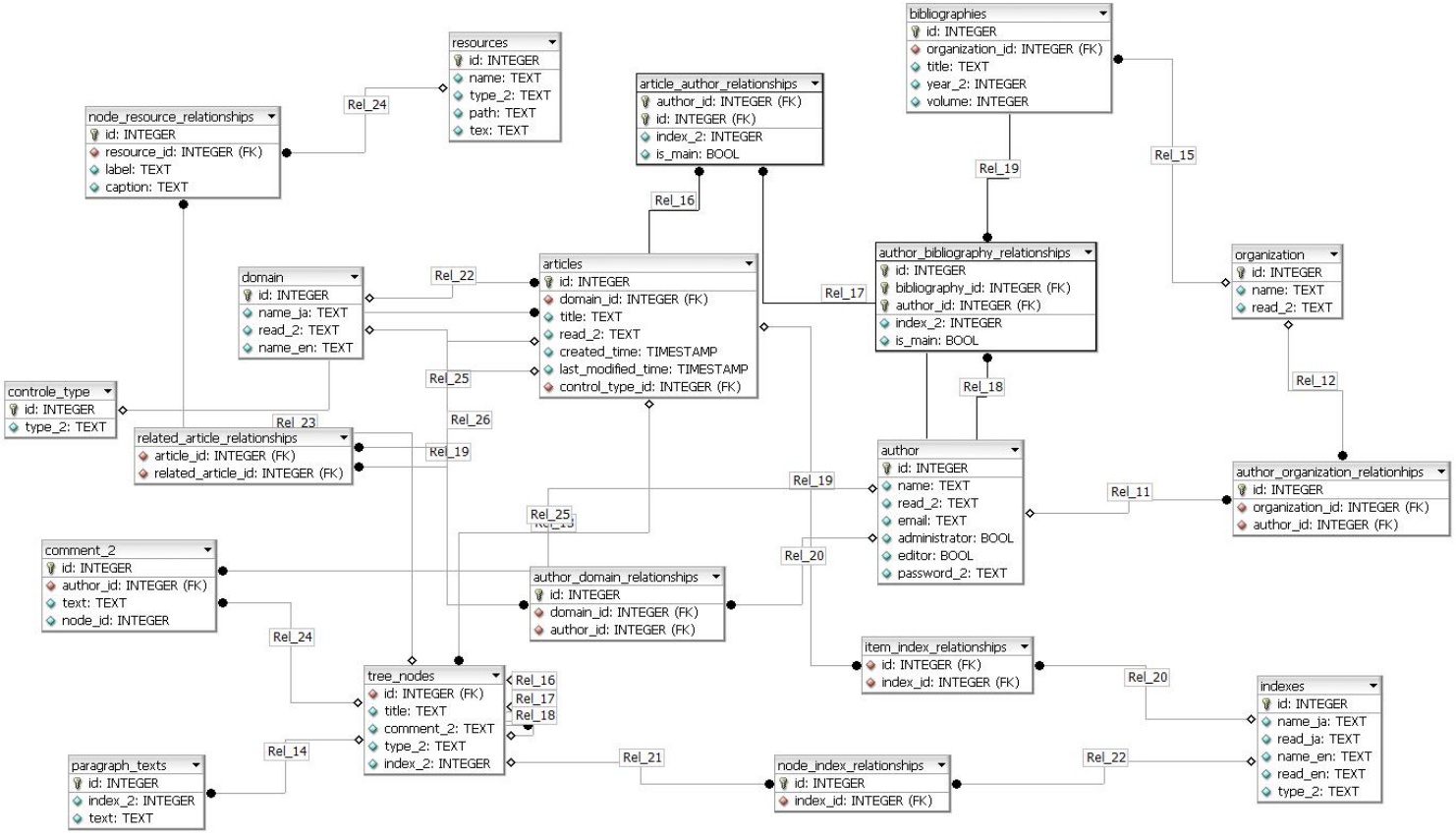
- このような書き方もありです
  - 友人帳テーブルが参照する外部テーブルの「趣味」属性の重複が気になる人はこうするとよいでしょう



より現実的なデータベース

あるWebサービスのデータベース

- 長尾研で開発中の人工知能学事典サーバーのデータベースの設計図
  - すべてのテーブルが他のテーブルを参照している



事典項目のテーブル

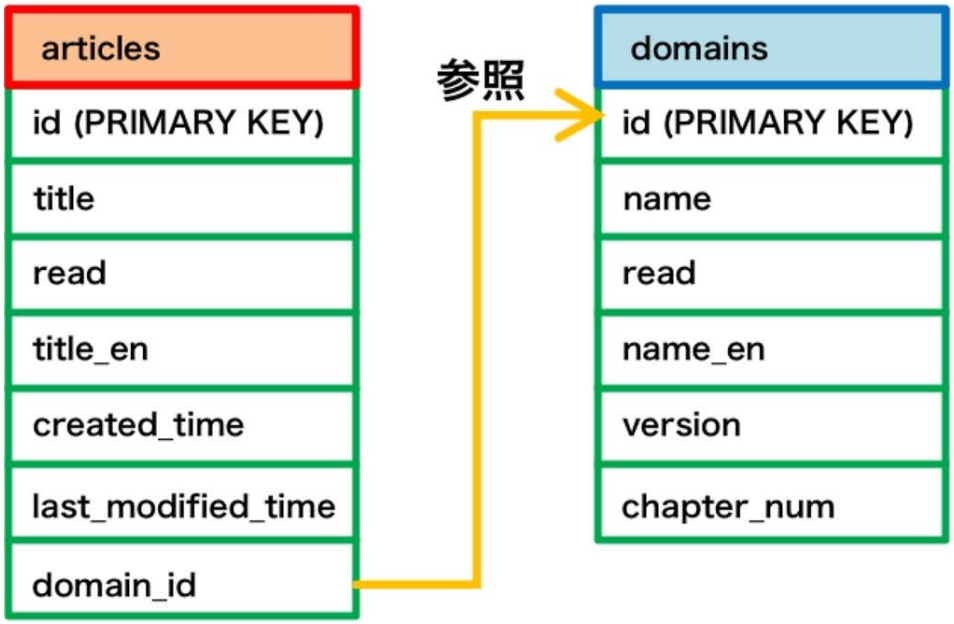
- CREATE TABLE articles (
  id serial NOT NULL PRIMARY KEY,
  title text, read text, title_en text,
  created_time timestamp,
  last_modified_time timestamp,
  domain_id integer, version integer,
  CONSTRAINT articles_domain_id_fkey
  FOREIGN KEY (domain_id)
  REFERENCES domains (id));
- 事典の項目はその分野（ドメイン）に依存する
- 分野に関する情報は、他のテーブル（domains）に記述されている

分野のテーブル

- CREATE TABLE domains (
  id serial NOT NULL PRIMARY KEY,
  name text, read text,
  name_en text, version integer,
  chapter_num integer);
- たとえば、「機械学習」という分野に、「深層学習」や「ニューラルネットワーク」という項目が含まれる

他のテーブルの参照

- 事典項目（articles）テーブルは分野（domains）テーブルを参照している
  - 分野（ドメイン）のidは、事典項目のdomain_idとして参照できる
  - つまり、項目のレコードから関連する分野の名前（name）などを調べることができる



サブクエリ

- SQLのSELECT文は、その条件部に
  同一あるいは異なるテーブルを参照する
  SELECT文を書くことができる
- SELECT * FROM articles WHERE domain_id =
  (SELECT id FROM domains WHERE name =
  ‘機械学習’);
  - 「機械学習」分野のすべての項目のレコードを
    表示する
- 特に、内側のSELECT文を内部クエリ、
  外側のSELECT文を外部クエリと呼ぶ

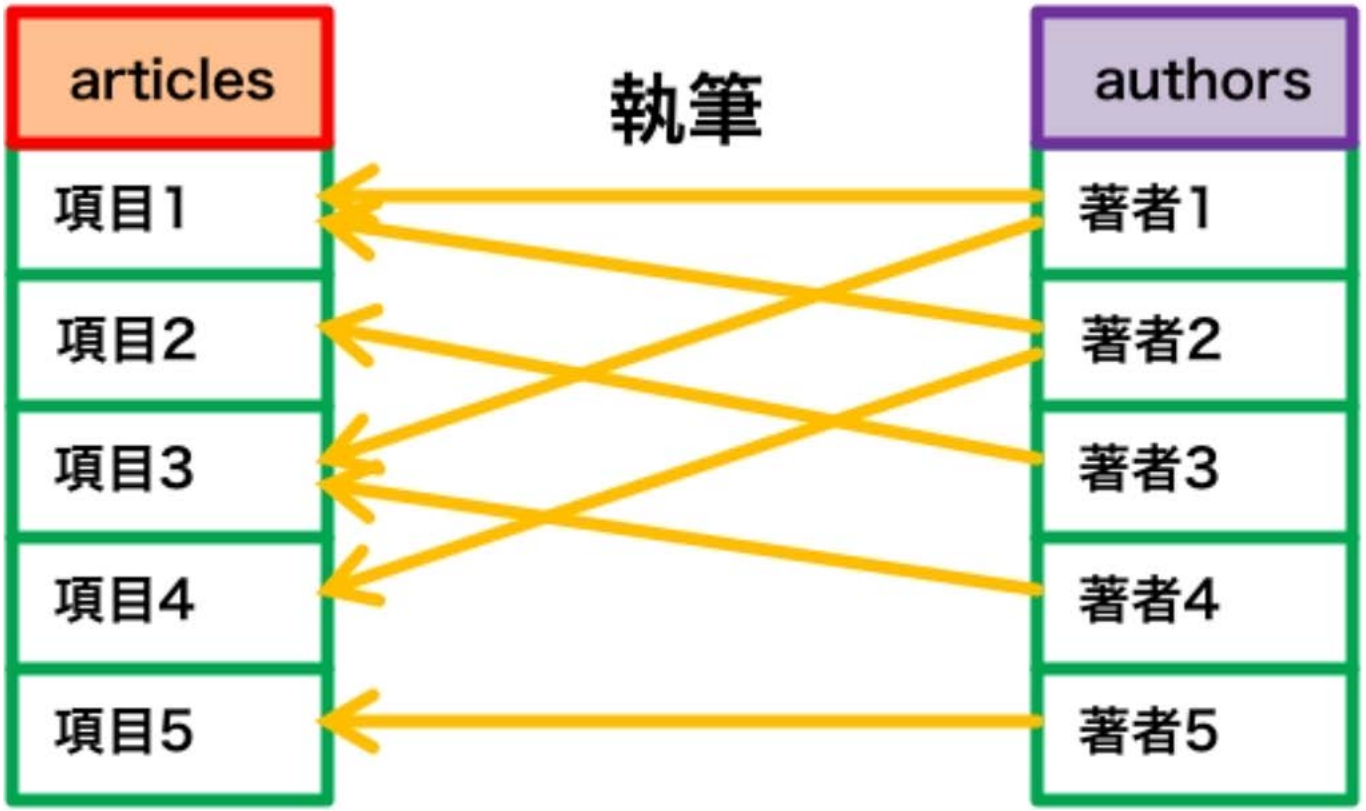
ところで著者情報はどこに？

- 事典にはそれを書いた人（著者）がいるはずだが、事典項目のテーブルには著者に関する属性がない
- ちなみに、著者に関するテーブル（authors）は以下のSQLで定義される

```
CREATE TABLE authors (
  id serial NOT NULL PRIMARY KEY,
  name text, read text, name_en text,
  email text, password text,
  is_editor boolean);
```

多対多の関係

- 複数の著者が一つの項目を書く場合がある
- 一人の著者が複数の項目を書く場合もある
- つまり、項目と著者は多対多の関係になる



多対多の関係の表現法

- 新たなテーブルarticle_author_relationships
- CREATE TABLE article_author_relationships (
  id serial NOT NULL PRIMARY KEY,
  article_id integer, author_id integer,
  CONSTRAINT article_author_relationships_article_id_fkey
  FOREIGN KEY (article_id) REFERENCES articles (id),
  CONSTRAINT article_author_relationships_author_id_fkey
  FOREIGN KEY (author_id) REFERENCES authors (id));
- 2つのテーブルのidを外部キーとして参照し、両者を関係づけることができる

複雑なサブクエリ

- 著者Aが書いたすべての項目を表示する
  - SELECT * FROM articles WHERE id IN (SELECT article_id FROM article_author_relationships WHERE author_id = (SELECT id FROM authors WHERE name = ‘A’));
- 項目Bの著者をすべて表示する
  - SELECT * FROM authors WHERE id IN (SELECT author_id FROM article_author_relationships WHERE article_id = (SELECT id FROM articles WHERE title = ‘B’));

ここで、ブレイク

講義に関するコメント(1/3)

- テーブルを正規化することで、一目で情報を取り出すことができ、とても便利だと感じました。部屋が整理されているのとそうでないのとでは、どこに何があるのかわかる速さに違いがあることに似ていると思いました。今の技術の中でこれが生かされている場面はどんなものがありますか。
- WHEREを使わないと大変なことになるとのことですが、どうしてUndoが使えないのですか？
- 自分の作った趣味の外部テーブルには読書、映画などが複数あってとても無駄が多いように思えるのですが、だからといって要素名を趣味にし、各人物に対応する属性を作成し、その属性のyesかnoで趣味かどうかを判別するのは人物が増えた際に趣味テーブルの属性を増やす必要があり、あまり拡張性がいいとは思えないのですが、何かいい方法はあるのでしょうか。

- 例えば、マイナンバーは国民の数だけレコードがあり、さらに住所や所得などのデータが追加されているので巨大なデータベースになります。
かなり大きなテーブルなので当然正規化されていると思いますが、その設計(スキーマ)を見てみたいですね。
- WHEREを使わないと、削除や変更の範囲を限定できませんし、一般にすべてのデータを一度に見ることはできませんので、想像の及ばない範囲でデータが変更される可能性があるのが問題なのです。
アンドゥについては、戻すべき範囲が大きすぎるとメモリがオーバーフローする可能性があるので一般に実装されていないことが多いです。事前にバックアップを取っておくのがよいでしょう。
- テーブルに同じ値(長い名前など)を何度も書くとメモリを圧迫するので、短い記号や数字に置き換えて、別のテーブルでその記号と正確な値を対応付けるというのが一般的なやり方です。
つまり、繰り返し使われることがわかっている名前にはあらかじめ略称を付けて、略称と正式な(長い)名前の対応表を作るというやり方です。

講義に関するコメント(2/3)

- データベースで使用頻度が高いデータにインデックスを用意して処理を高速化できるとありましたが、インデックスの量が大きくなった場合や、使用量に応じてアクセスの順位づけをしたい場合の方法にはどのようなものがあるのでしょうか。
- RDBには履歴を保存するのにあまり適さないという欠点がありますが、データウェアハウスなどをはじめとするNOSQLの良さがいまいち分かりません。どのように利用しているのですか？
- 以前からリレーショナルデータベースの勉強をしたいと思っていて、3年後期のデータベースの授業に合わせて自らデータベースを作成したいと思っていましたが、どのソフトがオススメですか？

- インデックスのサイズに応じた処理はSQLでは書けません。レコード数が大量になり検索効率が悪くなった場合の対処には、やはりテーブルを分けるなどの設計の見直しが必要になるでしょう。
- RDBは同じシステム上にデータがないと結合などの処理ができないので、分散環境に適していません。NoSQLはそれを解決するために考案されたものです。SQL的な処理を分散システムで利用できる(ただし、データ更新には時間がかかる)ようにしてビッグデータに対応しているのです。
- 私のおすすめはPostgreSQLというものです。pgAdminというツールがよくできています。あとはMySQLというソフトが有名です。いずれもフリーなので、ダウンロードして使ってみるとよいでしょう。また、プログラムからデータベースを呼び出すためのライブラリも豊富にあります。私は、C#やJavaのプログラムから、PostgreSQLを呼び出すというのをよくやっています。

講義に関するコメント(3/3)

- データベースを専門としている人たちはどんなテーマで研究しているのでしょうか？(1例でも教えて欲しいです！)
- Prologはインタプリタ言語で他の言語に処理を任せているとありましたが、SQLについても同様なのでしょうか。それとも独自の処理系をもっているのでしょうか。
- データベースを使いこなせる方が効率が良いと聞きましたが、先生はご自分が管理するデータはデータベースで管理されているのですか。(よく普通のPCにあるフォルダファイルとかではなく…？あまり良く分かっていないです。)もしそうであるなら方法はやはり自分でプログラムを組んで…ということでしょうか。
- 今、家に本があふれかえっていて、その本の情報を記録してまとめたいと思っています。しかし、スマホのメモ機能を使ってまとめた方が楽で簡単だと思います。どっちを使ったほうが良いですか？
- 最近では、Webでよく使われるJSONと呼ばれる構造化データを保存するNewSQLという仕組みが盛んに研究されているようです。今後データベースは間違いなく大規模化(ビッグデータ化)と分散化(ネットワーク化)に進んでいくと思います。
- SQLはデータベース管理システム(DBMS)上に処理系が実装されています。PostgreSQLやMySQLがその例です。
- さすがにファイルシステムをデータベースで置き換えるほどには使いこなしてはいません。しかし、Webサーバー上にアプリケーション(例えば、事典システム)を実装する場合は、ほぼ100%データベースを使います。
- 本のデータをAmazonからダウンロードして、Excelファイルに入力しておくとよいでしょう。100冊を超えるくらいから、データベースに移行すると検索が便利になります。

その他のコメント(1/4)

- JavaやC++といった言語を学びたいのですが、他の講義で少し触れたときにかなり難しく感じ、訳がわからなかったです。プログラミング言語の効率のよい学び方や、参考書などはありますか。
- 講義中で、C#が初心者にオススメだとおっしゃってましたが、先生が思う、他に知っておいた方が良い言語はどんなものでしょうか。まだ自分が何をやりたいか決まっていないのですが、どの分野でどんな言語が役に立つかが知りたいです。
- アルゴリズムで計算量を削っていくことがここ1年くらいの間、僕にとってアツいと感じているのですが、でもアルゴリズムは手段であって、目的とはなりえないものだと最近わかってきました。それでも何か、今までやってきたことをいかして仕事にしてやりたい気持ちがあります。何かアルゴリズムを考えることが仕事になった例をご存じないですか？

- プログラミング言語は何か一つでもしっかり覚えると他の言語も覚えやすくなります。プログラミング言語の本質はどれもそれほど違いはありません。しかしC++よりはJava、JavaよりはC#の方が初心者には向いています。さらに言えば、最近はPythonが覚えやすいと思います。参考書もありますので、必要なら言ってください。
- Windows上の一般的なプログラムを開発するなら、Visual StudioとC#を勉強するのがよいと思います。iPhoneのアプリを作りたいのならSwiftで。とにかく早く覚えて何かやりたいのならPythonがよいと思います。
- アルゴリズムで勝負するのは悪くないと思います。よいアルゴリズムを考えて、その実装プログラムを公開すれば、多くの人に感謝されるでしょう。古くは線形計画問題を高速で解くカーマーカー法、最近ではディープラーニングの高速化アルゴリズムが注目されています。

その他のコメント（2/4）

- VRなどの話が出ましたが、出力装置やネットワークインフラの技術の進捗はすさまじいのに、入力装置はキーボードとマウス（タッチスクリーンにしても仮想キーボード）のままいまいち進化していないような気がします。タイピングは訓練で速くなりますが、もっと思考をダイレクトに入出できるような装置や仕組みが出てこないでしょうか。脳波を機械学習して入力の補助にあてるとか出来ないでしょうか。
- VRグラスをこれ以上小型化することは可能でしょうか。今は重くて、利便性が悪いですが、小さくかつ軽くできたら、日常生活の中でも使えるようになると思います。
- VRのマルチユーザー（１対１や２対２ですが）として、レーザー銃とシールドをVR化して撃ち合いして点数を競うようなゲームがあるらしいです。実際にお互い横に移動したり、ジャンプしたりしたら座標計算もするそうです。未来的にはこういった本当のスポーツのようなe-sportsが世間一般にも（バスケ、サッカーなど）認知されていくのかなと思いました。
- 脳と機械をつなぐインタフェースをブレインマシンインタフェース（BMI）と呼びます。頭で考えたことがそのまま入力できたらすごいでしょうね。そのころには、頭にチップを埋め込むようになっているかも知れません。しかし、頭で考えたことをそのまま他人に見せるのはやめた方がよいでしょう。テレパシーなんて使えない方が世の中平和でしょうね。
- 今後は、VR/ARのためのハードウェアは小型で軽量なものになると思います。私は、スマホが変形してメガネ型になるデバイスを考えています。おそらくスマホが進化するとしたらそのような方向でしょう。
- VR/ARは人々の生活を大きく変えると思います。もちろん、スポーツも変わり、プロ（を模したAI）と試合をして、自分の実力を簡単に知ることができるようになると思います。早いうちに自分の才能に気づけるようになると、努力の甲斐があるかも知れませんね。

その他のコメント（3/4）

- VRの開発に興味があります。Unityでスマホアプリを作った経験もあったので、unityで開発してみようと思ったのですが、VRゴーグルなどが高く手が出せませんでした。実際に買うのは難しいかもしれませんが、おすすめのVRゴーグルなどはありますか？
- VRで触覚も再現できると言っていましたが、その場合、全身スーツのようなものを着なければならないのでしょうか？
- Gateboxを用いた疑似恋愛についていろいろ言ってましたが、VRを用いた恋愛の方がよっぽど危険そうに感じます。また、VRで触感まで再現されるかもと言って少し濁していましたが、VRで恋愛ができるようになり、触感までもしできるように....これって本当に危ないですね。
- すごく関係ないですけど、ユニティちゃんかわいいですよね。Unity内でユニティちゃんが動くソフトを作って遊んでました。先生は研究する立場になってから、遊びのソフトを作ったりしてますか。

- ちょっと値段が高いですが、HTC Viveがよいと思います。もし興味があるようなら、研究室に来て、試してみてはどうですか。
その気になれば、一週間くらいで簡単なVRゲームが作れるようになると思います。
- おそらく筋肉に直接、電気的な刺激を与えて、何かに触れているような感覚を作り出すのではないかと思います。その場合、スーツを着るというよりも、グローブをはめたり、腕に巻き付けたりするのだと思います。
- 同感ですね。しかし、そのような方向への進化はもはや止められないでしょう。私が昔、ニコ動を批判したのは、人間をダメにする危険性を感じたからです。テクノロジーが人間を救うのではなくダメにしていくのなら、何のために研究をしているのか疑問に思うことになるでしょう。
- 研究室の学生と共同でゲームやアトラクションのプログラムを作ったことがあります。ゲームを作るという経験は、プログラミングのためにとてもよいことだと思っています。
ところで、ユニティちゃんは3Dデータをダウンロードできるみたいなので、VR上で再現できそうです。

その他のコメント（4/4）

- 先日、ある記事で、有名企業社長や官公庁幹部ら各界で活躍する著名人が「いま22歳に戻れるなら、どの会社に入りたいか」というテーマでインタビューを受けているものを読みました。もし、長尾先生は今、22歳にもどれるとしたら、どの会社に入社したいですか。また、なぜですか。
- 先生は講義中にAmazonが嫌いだとか、嫌いなものについてよく述べられますが、好きなものは何でしょうか。
- 先生はプログラミングバイトをしていたとおっしゃっていましたが、いつからプログラをしはじめたのですか？そして独学ですか？私は女の子をきせかえするプログラゲームですら途中であきらめてしまったのに、それを商売にしているなんて尊敬です。やはりできる人は昔から違うんですね。赤ちゃんから出直したいです。
- 長尾先生はたくさんのコメントにすごく真剣に回答されています。どういった心境なのでしょうか。純粋に気になりました。
- すぐに思いつくのはマイクロソフトですね。最近とてもセンスがいいからです。しかし、日本には研究所がないので、アメリカに行くことになると思います。グーグルは面白い会社だと思いますが、創業者たちがどうもうさんくさい気がするので入社したいとは思わないです。アップルはスティーブ・ジョブズが生きているうちに入ってみたかったですね。
- アマゾンは日本に法人税を払ってくれれば好きになると思います。最近好きなのはマイクロソフトですが、それ以外には、パロットというフランスにあるドローンの会社が好きです。
- プログラミングを最初にやってのは高校生のときで、言語はBASICです。大学生のときにアルバイトでプログラミングをやっていたときは、C言語でした。しかし、今のように大学で教わったのではなく、独学でした。プログラミングは大好きです。やはり、好きこそものの上手なれ、ということでしょう。
- やはり、みなさんとコミュニケーションをしたいと思っています。みなさんには無限の可能性がありますから、何とかしてその可能性を伸ばしてあげられないかといつも思っています。

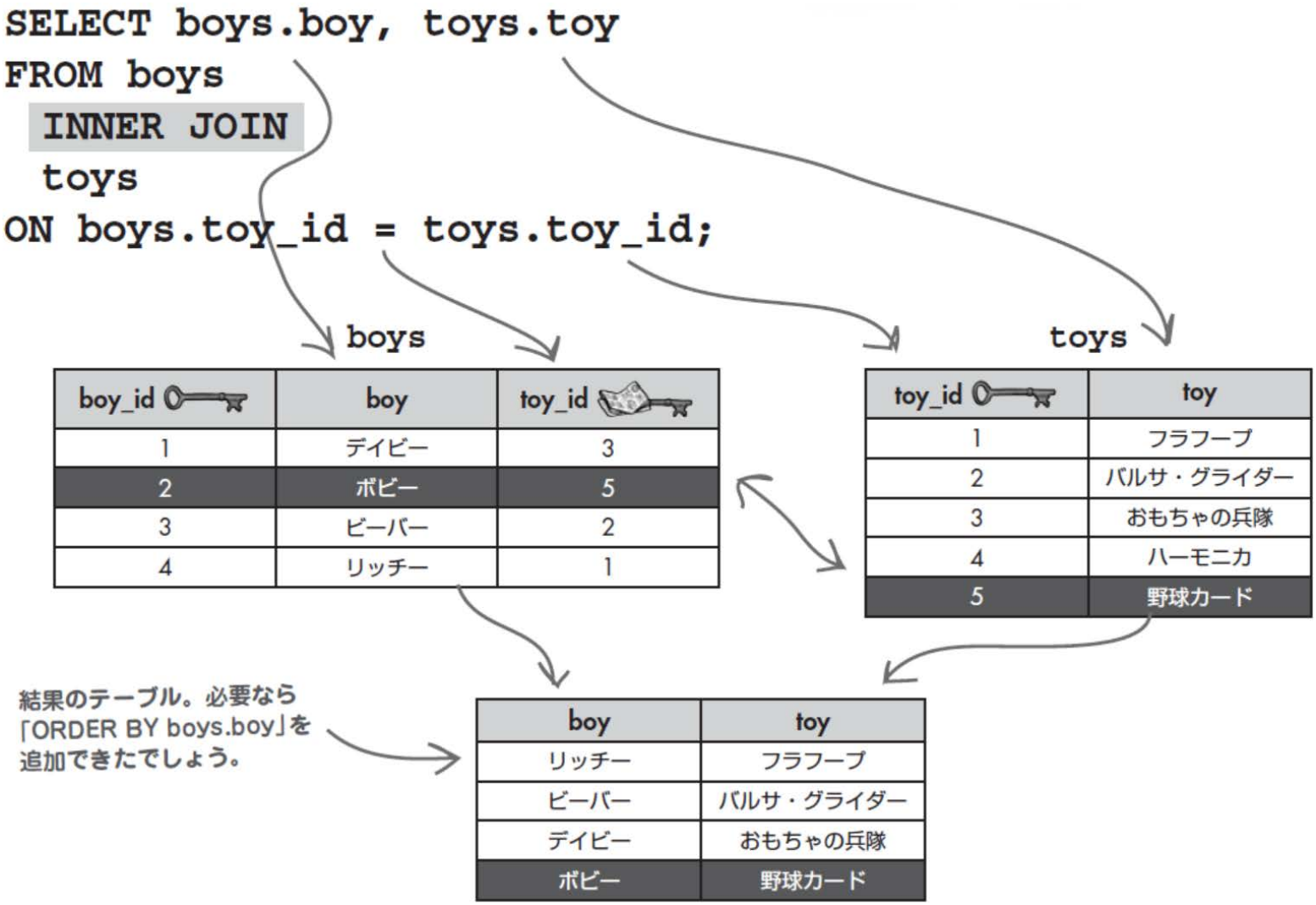
テーブルの結合（データベースにおける演算）

テーブルの結合（1/5）

- データベースのテーブルは行列（matrix）のようにも見える
- 行列には演算（加法や乗法）が定義されている
- テーブルにもそのような手続きが見つかるはず
  - それが結合（join）
- 結合には、内部結合と外部結合がある
  - 内部結合（inner join）：2つのテーブルの合致する部分を返す（集合のintersectionのようなもの）
  - 外部結合（outer join）：2つのテーブルのすべての組合せを返す（集合のunionのようなもの）

テーブルの結合（2/5）

- 内部結合（inner join）



テーブルの結合（3/5）

- 自然結合（natural join）

```
SELECT boys.boy, toys.toy
FROM boys
  NATURAL JOIN
  toys;
```

boys

boy_id	boy	toy_id
1	デイビー	3
2	ボビー	5
3	ビーバー	2
4	リッチー	1

toys

toy_id	toy
1	フラフープ
2	バルサ・グライダー
3	おもちゃの兵隊
4	ハーモニカ
5	野球カード

等価結合（最初に紹介した内部結合）を使った時と同じ結果集合が得られます。

boy	toy
リッチー	フラフープ
ビーバー	バルサ・グライダー
デイビー	おもちゃの兵隊
ボビー	野球カード

NATURAL JOIN
内部結合は、合致する列名を識別します。

テーブルの結合（4/5）

- 左側外部結合（left outer join）

```
SELECT g.girl, t.toy
FROM toys AS t
LEFT OUTER JOIN girls AS g
ON g.toy_id = t.toy_id;
```

左側のテーブル

右側のテーブル

toys

toy_id	toy
1	フラフープ
2	バルサ・グライダー
3	おもちゃの兵隊
4	ハーモニカ
5	野球カード
6	ティンカー・トイ
7	エッチ・ア・スケッチ
8	スリンキー

girl_id	girl	toy_id
1	ジェイン	3
2	サリー	4
3	シンディ	1

テーブルの順序を変更すると、
このようなテーブルが得られます。

girl	toy
シンディ	フラフープ
NULL	バルサ・グライダー
ジェイン	おもちゃの兵隊
サリー	ハーモニカ
NULL	野球カード
NULL	ティンカー・トイ
NULL	エッチ・ア・スケッチ
NULL	スリンキー

合致するものが見つかった場合、
それがテーブルに結果として現れます。
合致するものが見つからなかった場合、
テーブルに行は現れますが、
その値はNULLになります。

テーブルに列が現れる
順序は、SELECT した
順序です。この順序は、
左側外部結合とは
何の関係もありません。

テーブルの結合（5/5）

- 右側外部結合（right outer join）

```
SELECT g.girl, t.toy
FROM toys AS t ←左側のテーブル
RIGHT OUTER JOIN girls AS g ←右側のテーブル
ON g.toy_id = t.toy_id;
```

右側のテーブル

girl

girl_id	girl	toy_id
1	ジェイン	3
2	サリー	4
3	シンディ	1

左側のテーブル

toys

toy_id	toy
1	フラフープ
2	バルサ・グライダー
3	おもちゃの兵隊
4	ハーモニカ
5	野球カード
6	ティンカー・トイ
7	エッチ・ア・スケッチ
8	スリンキー

この場合は内部結合（inner join）と同じ結果になる

結果

girl	toy
シンディ	フラフープ
ジェイン	おもちゃの兵隊
サリー	ハーモニカ

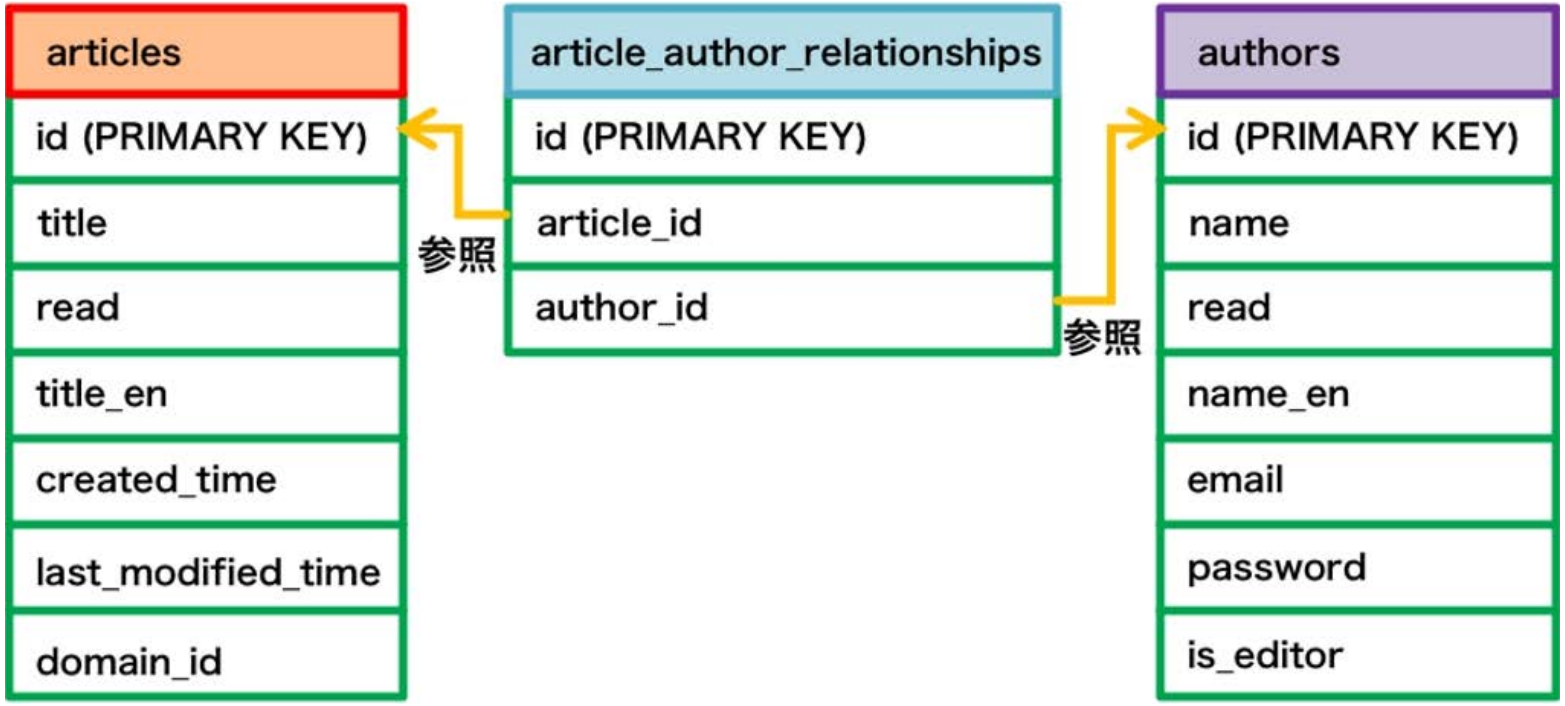
結合は非常に便利

- SELECT a.title, d.name FROM articles AS a INNER JOIN domains AS d ON a.domain_id = d.id;
  - 事典項目とその分野の一覧（Aとする）
- SELECT a.title, d.name FROM articles AS a LEFT OUTER JOIN domains AS d ON a.domain_id = d.id;
  - Aの情報＋分野の決まっていない項目一覧
- SELECT a.title, d.name FROM articles AS a RIGHT OUTER JOIN domains AS d ON a.domain_id = d.id;
  - Aの情報＋項目を含んでいない分野一覧

事典の例で
結合を使ってみよう

事典項目と著者の関係

- 両者の関係は多対多で、それぞれのテーブルには直接の接点が何もない
- そこで両者を仲介するテーブルを作る



内部結合で
項目と著者の
対応表を見る

内部結合で項目と著者の対応表を見る

- 項目タイトルと著者IDの対応を調べるSQL
- SELECT a.title, r.author_id FROM articles AS a INNER JOIN article_author_relationships AS r ON a.id = r.article_id;
- もともとの事典項目には著者の情報は含まれていない
  - 結合によって著者の情報（この場合はID）が追加された

外部結合で
著者が決まって
いない項目も含めた
対応表を見る

外部結合で著者が決まっていない項目も含めた対応表を見る

- 項目タイトルと著者IDの対応（対応しない項目を含む）を調べるSQL
- SELECT a.title, r.author_id FROM articles AS a LEFT OUTER JOIN article_author_relationships AS r ON a.id = r.article_id;
- 内部結合と異なり、著者が決まっていない項目も表示される（著者IDはNULLになる）
  - 著者が誰かということだけでなく、著者が決まっていない項目も一覧できる

結合を繰り返す

結合を繰り返す

- サブクエリを使って結合を繰り返し
  項目タイトルと著者名の対応を調べるSQL
- SELECT a1.title, a2.name FROM authors AS a2
  INNER JOIN (SELECT * FROM articles AS a
  INNER JOIN article_author_relationships AS r
  ON a.id = r.article_id) AS a1
  ON a2.id = a1.author_id;
- articlesとarticle_author_relationshipsの内部結合の結果とauthorsを内部結合する

まとめ

- データベースの複数のテーブルを組み合わせる
  - 正規化されたテーブルは組合せが容易
  - サブクエリ
    - SELECT文の条件に他のSELECT文を書ける
    - 内側のSELECT文を内部クエリ、外側を外部クエリと呼ぶ
- 結合
  - データベース上に定義される演算
  - 内部結合と外部結合
    - 外部参照しているテーブルと元のテーブルをまとめる
    - 属性が合致するもののみを集めるのが内部結合、合致しないものも集めるのが外部結合
  - 結合の繰り返し
    - サブクエリを用いた結合の入れ子
    - 多対多の関係も一つにまとめることができる

参考図書

- Head First SQL
  - 電子書籍
  - 読んでみたい人は私に連絡してください

講義資料はダウンロードできます

- URLは、以下の通り。
  - http://www.nagao.nuie.nagoya-u.ac.jp/syllabus/knowledge_proc.html
- メールでの質問も受け付けます。
  研究室に遊びに来たい人は連絡ください。
  今月末(5/30)に見学会を企画しています。
  - nagao@nuie.nagoya-u.ac.jp

今日の問題

- 右下の2つのテーブルarticlesとdomainsに関して、次の2つのSELECT文の検索結果を書いてください。
  - 内部結合

```
SELECT a.title, d.name
FROM articles AS a
INNER JOIN domains AS
d ON a.domain_id=d.id;
```

articles

id	title	domain_id
1	フレーム	1
2	ニューラルネット	2
3	ロボット	3
4	将棋	4
5	囲碁	4
6	結合	5

domains

id	name
1	知識表現
2	機械学習
3	ロボティクス
4	ゲーム
10	応用

  - 左外部結合

```
SELECT a.title, d.name
FROM articles AS a
LEFT OUTER JOIN domains AS d ON a.domain_id=d.id;
```

参考

- **UNITY-CHAN! OFFICIAL WEBSITE**
  - **http://unity-chan.com/**