リレーショナルDBにおける複数のテーブル

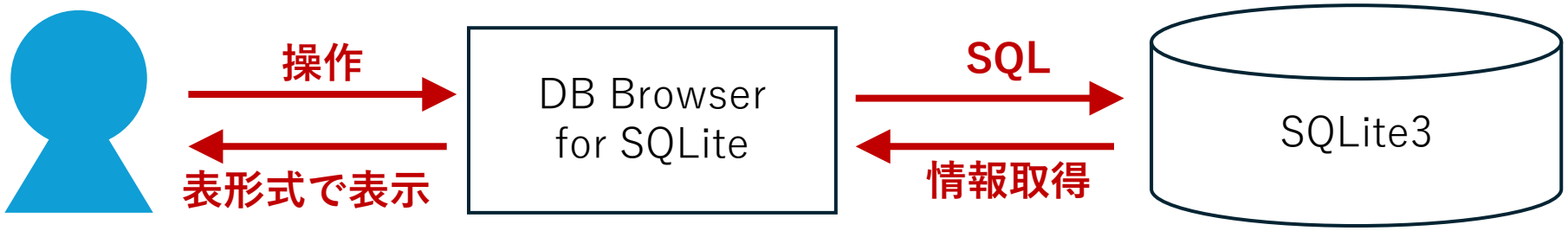
国立情報学研究所学術コンテンツ課
瀬尾崇一郎

初日の内容振り返り

- DBMSというのがあるらしい。
  - リレーショナルDBというのがメジャーらしい。
- 「DB Browser for SQLite」を使うと使えるらしい。
  - Excelみたいに表示できるらしい。
- 「SQL文」というもので検索や操作ができるらしい。

→「SQL文」で操作できるExcelみたいなもの？

「DB Browser for SQLite」の特性



- SQlite3というリレーショナルDBの中身を、
  Excelのような表計算ソフトっぽいガワで見せてくれる。

- Excelの表のようにSQLite3内のテーブルを閲覧・修正できる。

→「1テーブルだけを扱ってる」限りにおいては、
だいたいExcelのような使用感になる

「テーブル」とは？

- リレーショナルDBを構築する要素。
- 複数の列（項目）を設定し場所に、１行１データで
  情報を登録していく、表形式の保管場所。
- リレーショナルDBでは、複数のテーブルを扱うらしい。

「テーブル」とは？

- あなたは『大学図書館の貸出記録』を、Excelで管理することになりました。
  → きっとこんなExcel表を作るのでは？

書名	貸出相手名	貸出日	貸出相手学籍番号	返却日	購入日	貸出相手所属
SQLポケットリファレンス	情報太郎、情報花子	2025/6/10、2025/7/5	20250001、20240010	2025/6/17、	2025/4/12	情報学部、情報学部
SQLポケットリファレンス（２冊め）					2025/5/10	

「テーブル」とは？

- リレーショナルDBでは、以下のような複数テーブルで管理する

所蔵テーブル		
所蔵ID	書名	購入日
1	SQLポケットリファレンス	2025/4/12
2	SQLポケットリファレンス	2025/5/10

利用者テーブル			
利用者ID	氏名	学籍番号	所属
1	情報太郎	20250001	情報学部
2	情報花子	20240010	情報学部

貸出記録テーブル				
貸出記録ID	所蔵ID	利用者ID	貸出日	返却日
1	1	1	2025/6/10	2025/6/17
2	1	2	2025/7/5	

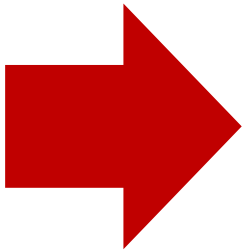
複数テーブルへのSQL

- これがExcelだったら、「複数テーブル」をどう扱う？
  →　画面表示シートを切り替えながら把握する？
  - Excelの良いところは一覧性
    →　複数シートを扱うのは一覧しづらい。つらい（私見）

- リレーショナルDBでは、SQLによって一度に扱うことができる。

複数テーブルへのSQL

```
SELECT
        所蔵テーブル.書名,
        利用者テーブル.氏名,
        貸出記録テーブル.貸出日
FROM
        所蔵テーブル,
        利用者テーブル,
        貸出記録テーブル
WHERE
        所蔵テーブル.所蔵ID = 貸出記録テーブル.所蔵ID
            AND
        貸出記録テーブル.利用者ID = 利用者テーブル.利用者ID;
```



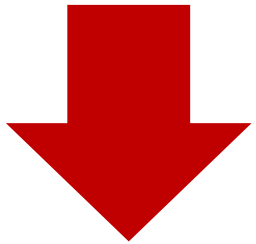
所蔵テーブル.書名	利用者テーブル.氏名	貸出記録テーブル.貸出日
SQLポケットリファレンス	情報太郎	2025/6/10
SQLポケットリファレンス	情報花子	2025/7/5

複数テーブルへのSQL

所蔵テーブル		
所蔵ID	書名	購入日
1	SQLポケットリファレンス	2025/4/12
2	SQLポケットリファレンス	2025/5/10

利用者テーブル			
利用者ID	氏名	学籍番号	所属
1	情報太郎	20250001	情報学部
2	情報花子	20240010	情報学部

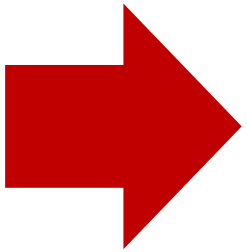
貸出記録テーブル				
貸出記録ID	所蔵ID	利用者ID	貸出日	返却日
1	1	1	2025/6/10	2025/6/17
2	1	2	2025/7/5	



所蔵テーブル.書名	利用者テーブル.氏名	貸出記録テーブル.貸出日
SQLポケットリファレンス	情報太郎	2025/6/10
SQLポケットリファレンス	情報花子	2025/7/5

複数テーブルへのSQL（JOIN派）

```
SELECT
        所蔵テーブル.書名,
        利用者テーブル.氏名,
        貸出記録テーブル.貸出日
FROM
        ( 所蔵テーブル
                INNER JOIN
          貸出記録テーブル
           ON 所蔵テーブル.所蔵ID = 貸出記録テーブル.所蔵ID )
                INNER JOIN
        利用者テーブル
         ON 貸出記録テーブル.利用者ID = 利用者テーブル.利用者ID;
```



所蔵テーブル.書名	利用者テーブル.氏名	貸出記録テーブル.貸出日
SQLポケットリファレンス	情報太郎	2025/6/10
SQLポケットリファレンス	情報花子	2025/7/5

複数テーブルの考え方

- リレーショナルDBでは、複数のテーブルを扱うらしい。
- 複数のテーブルを扱うときには、SQLを使うらしい。

- 「複数のテーブル」を作るときには、どういう考え方をすればいい？

「テーブル」の基本

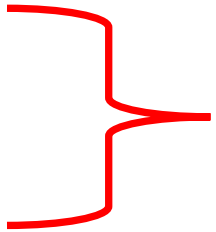
- RDBのテーブルには、基本的に「主キー」が必要
  - 主キー：テーブル内の各レコードを一意に区別できる情報
  - 「書名」と「購入日」のセットで判別するという手もあるか？
    →だとすると、同じ「購入日」に同じ書名を2冊買ってはいけない？
  - 所蔵本1冊ごとに「所蔵ID」を追加し、これを主キーとする。

所蔵ID（主キー）	書名	貸出相手名	貸出日	貸出相手学籍番号	返却日	購入日	貸出相手所属
1	SQLポケットリファレンス	情報太郎、情報花子	2025/6/10、2025/7/5	20250001、20240010	2025/6/17、	2025/4/12	情報学部、情報学部
2	SQLポケットリファレンス（２冊め）					2025/5/10	

データベースの「正規化」

- 「データベース工学」は情報学研究の一分野であり、リレーショナルDBが考案されて以後（1970年代）、どのような作り方をすると良いか様々な研究がされた。

- 「データベースの正規化」
  - 第一正規形
  - 第二正規形
  - 第三正規形

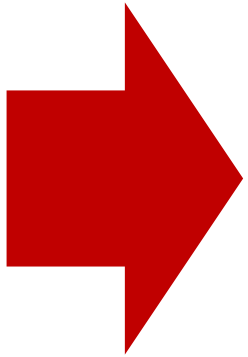
  特に重要
  - ボイス・コッド正規形
  - 第四正規形
  - 第五正規形

データベースの「正規化」

- 第一正規化
  - 同じ情報が１データ中に繰り返し出現するのはよくない。

書名	貸出相手名
SQLポケットリファレンス	情報太郎、情報花子

書名	貸出相手名１	貸出相手名２
SQLポケットリファレンス	情報太郎	情報花子



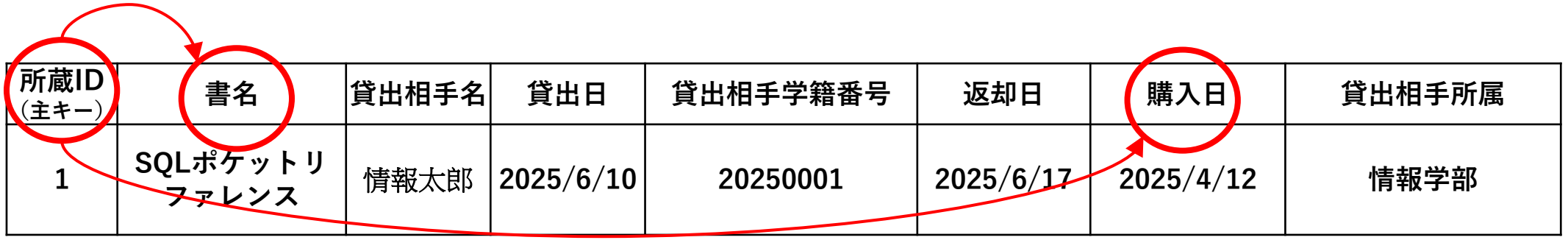
書名
SQLハンドブック

貸出相手名
情報太郎
情報花子

データベースの「正規化」

- 第二正規化
  - 「部分関数従属」を排除する
- 第三正規化
  - 「推移的関数従属」を排除する

- １テーブル内の情報は、そのテーブルの「主キー」に従うべき。
  - 「主キー」以外の列に伴うなら、その列と一緒に分けるべき。
  - 「主キー列」→「A列」→「B列」みたいな推移的な従い方も、分けるべき
- 「所蔵ID（この本を所蔵したこと）」がこのテーブルの主キーなら、これに従っていると言えるのは、「書名」と「購入日」。



所蔵ID（主キー）	書名	貸出相手名	貸出日	貸出相手学籍番号	返却日	購入日	貸出相手所属
1	SQLポケットリファレンス	情報太郎	2025/6/10	20250001	2025/6/17	2025/4/12	情報学部

データベースの「正規化」

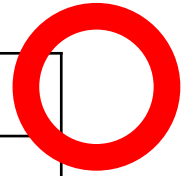
- 「貸出相手（利用者）」にまつわる情報は、「所蔵ID」に従うものではない。
- 「この本を」「この人に」貸し出した、という情報も、「所蔵ID」単独に従うものではない。

所蔵ID（主キー）	書名	貸出相手名	貸出日	貸出相手学籍番号	返却日	購入日	貸出相手所属
1	SQLポケットリファレンス	情報太郎	2025/6/10	20250001	2025/6/17	2025/4/12	情報学部

所蔵ID（主キー）	書名	購入日
1	SQLポケットリファレンス	2025/4/12

貸出相手名	貸出相手学籍番号	貸出相手所属
情報太郎	20250001	情報学部

貸出日	返却日
2025/6/10	2025/6/17



RDB的なテーブル構成

- それぞれのテーブルにID列を作って主キーを設定
- 「貸出記録」は、「誰が（利用者）」「何を（所蔵）」を各テーブルから参照
- 「所蔵ID」「利用者ID」だけだと、同じ人間が同じ本を何度も借りたときに区別できないので「貸出記録ID」を用意
  - こういうのを「サロゲートキー（代理キー）」と言ったりする。

所蔵テーブル		
所蔵ID（主キー）	書名	購入日
1	SQLポケットリファレンス	2025/4/12
2	SQLポケットリファレンス	2025/5/10

利用者テーブル			
利用者ID（主キー）	氏名	学籍番号	所属
1	情報太郎	20250001	情報学部
2	情報花子	20240010	情報学部

貸出記録テーブル				
貸出記録ID（主キー）	所蔵ID	利用者ID	貸出日	返却日
1	1	1	2025/6/10	2025/6/17
2	1	2	2025/7/5	NULL

「インデックス」とは？

- テーブルに保存したレコードについて、指定した列に「インデックス」を作る設定ができる

- インデックスを作った列に対しては、検索や取得の高速化が可能。
  - 事前に一通り調べて、メモを貼っておくようなイメージ。

- 代わりに、インデックスが設定されてるとデータ登録時には余計に時間がかかり、またデータ量も増大する。
  - 「あらかじめメモを貼っておく」作業の分だけ余計な仕事が増える。
  - 「貼ったメモ」の枚数分だけファイルの厚み重さが増える。

「インデックス」の考え方

- 全テーブルの全ての列にインデックスを作るようなことは、RDBにおいては基本的に非現実的。

- 主な使い方＝想定されているSQLにて、検索条件に使用される列に作っておくと、処理高速化の恩恵を受けられる。
  - 各テーブルのID列、参照・結合に使うID列
  - その他、「検索」に使う情報の列。
    - 「いついつ以降の貸出記録」→「貸出日」列

所蔵テーブル

所蔵ID（主キー）	書名	購入日
1	SQLポケットリファレンス	2025/4/12
2	SQLポケットリファレンス	2025/5/10

利用者テーブル

利用者ID（主キー）	氏名	学籍番号	所属
1	情報太郎	20250001	情報学部
2	情報花子	20240010	情報学部

貸出記録テーブル

貸出記録ID（主キー）	所蔵ID	利用者ID	貸出日	返却日
1	1	1	2025/6/10	2025/6/17
2	1	2	2025/7/5	

複数のテーブルとは

- リレーショナルDBのテーブル構造を考えることとは、
  取り扱うデータのことについて、あらかじめよく考えること。
  →対象の「世界を整理すること」。

- インデックスを作るとは、作ったDBの使われ方について、
  あらかじめよく考えること。
  →対象の「未来を整理すること」。

- 逆に言えば、「あらかじめ考えて」はいなかったものについて
  後から追加で扱おうとすると、いちいち大工事になりがちなのが
  リレーショナルDB。

標準ID情報の価値

- たとえば、「ISBN」情報があるとどんなことができるか？

- ISBN：「国際標準図書番号」
  - （10桁と13桁があったりと、色々ある）

- ISBNは、個々の「本」が持つ情報
  - このテーブル構成であるなら、「所蔵テーブル」が持つ情報と考えるのが妥当。

所蔵テーブル			
所蔵ID（主キー）	書名	購入日	ISBN
1	SQLポケットリファレンス	2025/4/12	9784774187327
2	SQLポケットリファレンス	2025/5/10	9784774187327

利用者テーブル			
利用者ID（主キー）	氏名	学籍番号	所属
1	情報太郎	20250001	情報学部
2	情報花子	20240010	情報学部

貸出記録テーブル				
貸出記録ID（主キー）	所蔵ID	利用者ID	貸出日	返却日
1	1	1	2025/6/10	2025/6/17
2	1	2	2025/7/5	

標準ID情報の価値

利用者テーブル			
利用者ID	氏名	学籍番号	所属
1	情報太郎	20250001	情報学部
2	情報花子	20240010	情報学部

所蔵テーブル			
所蔵ID（主キー）	書名	購入日	ISBN
1	SQLポケットリファレンス	2025/4/12	9784774187327
2	SQLポケットリファレンス	2025/5/10	9784774187327

貸出記録テーブル				
貸出記録ID	所蔵ID	利用者ID	貸出日	返却日
1	1	1	2025/6/10	2025/6/17
2	1	2	2025/7/5	

（外部DB）

書名	出版年月日	ISBN
SQLポケットリファレンス改訂第4版	2017.03	978-4-7741-8732-7

より詳しくは…

参考資料

図書貸出システムのデータベースを設計してみる