2019年9月24日　東京大学・前田

大学図書館員のためのIT総合研修 2019 実習シナリオ

実習環境

DB Browser for SQLite: https://sqlitebrowser.org/
住所.jp: http://jusyo.jp/
大学図書館員のための図書館システム開発練習用データ: https://mbc.dl.itc.u-tokyo.ac.jp/data4librarysystem/

1日目午後

1. データベースを立ち上げてみる

No	項目	作業内容	理解のポイント
1	DB起動	DB Browserを起動。[Open Database]メニューで、"住所.jp"のファイルを開く	リレーショナルデータベース管理システムと、データファイルは別であることを知る
2	DB確認	[Brows Dataタブ]で住所.jp のデータが表形式になっていることを確認	データベースは表形式であることを知る
3		town_memoの列に「NULL」が入っていることを確認する	リレーショナルデータベースには、Excel等スプレッドシートとは違う「お約束」があることを知る。 ただし、「NULL」（値がない）の使用は実はリレーショナルデータベースでは推奨されていない。

2. DB Browserの機能でデータベース操作

No	項目	作業内容	理解のポイント
4-A	テーブル確認	[Tableプルダウン]で ad_addressとt_versionの表の切り替えができることを確認する。	データベースは複数の表からなることを知る
5-A		office_flgのフィルタ列に"1"を入れ絞り込み、office_nameがすべて事務所になることを確認する	レコードの状況を示すフラグを活用することを知る
6-A	検索	フィルタで都道府県名を「北海道」に限定する	検索はフィールドごとの処理になる
7-A	レコード作成	新規レコードを追加する。表の末尾に初期値がすべてNULLのレコードができるので、そこに値をいれる	idが自動採番されることに言及　→　リレーショナルデータベースではデータの特定のためにユニークキーが重要
8-A	レコード削除	レコードの削除を行う。カーソルを削除するレコードに合わせ[Delete Record]ボタンをクリック（マウス右ボタンクリックから、プルダウンメニューの[Delete Record]でもよい）	表のdelete_flgに着目。論理的に削除するようなテクニックがあることも言及する
9-A	レコード修正	既存のレコードを修正する。	Excel等スプレッドシートと同じようにセルの中の値を書き換えられる。
10-A	テーブル構造の変更	「jinko」（人口）列の追加。テーブルクリック　→　modify tableで既存の表の枠組みを変えられる	データ型や制約を細かく設定できることを知る。人口なので「数値」(Integer)がよい。
11	Export	フィルタで都道府県名を「北海道」に限定する。フィルタで絞ったレコードをExportする。 [Save the tables as a currently displayed]アイコンで、CSV（実際にはTSV)で保存	データベースファイルは別形式のデータに変換できる
12	Import	上記のレコードをImportする　[New Database]で新DB作成　→　テーブル作成キャンセル　→ Import	データベースは指定の形式のデータを取り込むことができる。ただし、CSVにはデータ型がないので、その手当は必要

3 . SQLでデータベース操作

No	項目	作業内容	ポイント
4-B	テーブル確認	[Excute SQL]タブから以下のとおり、SQLコマンドを記載して▶ボタンで実行する。 DB中に「ad_address」と「t_version」という2つのテーブルがあることを確認できる。 *SELECT name FROM sqlite_master WHERE type='table';*	SQLite3固有の確認方法。DBシステムによって確認方法は異なる。
5-B	検索	[Excute SQL]タブから以下のとおり、SQLコマンドを記載、コマンドをマウスで選択し▶ボタンで実行する。 office_nameがすべて事務所になることを確認する。 *SELECT * FROM ad_address WHERE office_flg='1';*	select文の基本構文を確認する **select** *　→　selectに続けて出力フィールド指定。「*」は全フィールド **from** *ad_address*　→ from に続けてテーブル名指定 **where** *office_flg='1'*　→ where句以下で検索条件を指定
6-B		県名で検索する。以下は「北海道」を検索する例 完全一致検索例: *SELECT * FROM ad_address WHERE ken_name='北海道';* 前方一致検索例: *SELECT * FROM ad_address WHERE ken_name LIKE '北海%';*	文字列検索には、完全一致と中間一致を使分けることができる
7-B	レコード作成	画面右下のDB Schemaパネルから、ad_addressをダブルクリックし、テーブル構造を確認 以下の例のようなSQLでデータ登録をしてみる [Brows Data]タブで表示される一覧の一番上に、登録したレコードが追加されたことを確認する。 例： INSERT INTO ad_address (id, zip, ken_name) VALUES('89991', '1234-5678', '北海道');	ad_addressテーブルの「id」はテーブルのユニークキー（必須・重複不可）であることに注目
8-B	レコード削除	レコード作成実習で作成したレコードを削除する。まず、SELECT文で以下の例のように削除対象レコードを確認し、 例： *SELECT * FROM ad_address WHERE id='89991';* 同じ条件(WHERE句）で削除を実行する。 例： *DELETE FROM ad_address WHERE id='899991';*	WHERE句の条件の間違いに注意。
9-B	レコード修正	「北海道」のヨミを「ホッカイドウ」から「ホッカイドー」に一括変更する *UPDATE ad_address SET ken_furi='ホッカイドー' WHERE ken_id='1';*	GUIではできなかった一括処理がSQLなら可能なことを知る
10-B	テーブル構造の変更	「kencho」（県庁）列の追加 *ALTER TABLE ad_address ADD kencho text;*	データ型や制約を細かく設定できることを知る

2日目午前

4 .テーブルの作成・組み合わせ処理・集計

No	項目	作業内容	ポイント
11	テーブルの作成(GUI)	県知事テーブルを作る １．[Databae Structure]タブ→[Create Table]ボタン→[Edit Table Definition]ウインドウが開く ２．[Edit Table Definition]ウインドウのtableにテーブル名"chiji"を入れる ３．[Edit Table Definition]ウインドウの[Add Field]ボタンでフィールド設計をする 　外部連携用の[ken_id]と、知事氏名用の[name]を設定する	ad_addressに"chiji"（知事）列を追加するより合理的かどうかについて考えてみる ken_nameではなく、ken_idをキーにするのがお薦め
12	テーブルの削除(GUI)	[Databae Structure]タブからテーブル"chiji"を選び、[Delete Table]ボタンをクリックする	
13	テーブルの作成(SQL)	SQLでテーブルを作る。以下の例を拡張し、リッチな項目をもったテーブルを作成すること。 最低限の項目を指定した例: *CREATE TABLE chiji (ken_id integer, name text);*	データベースの作成には各フィールドのデータ型（数値、テキスト）の設計がポイント 同姓同名の知事の混同を避けるにはどうするのがよいかも考えてみると面白い
14	デーブルにデータ投入	インターネットにある知事のデータを、[Brows Data]タブの画面からデータを投入する。 （知事データには、たとえば　http://www.nga.gr.jp/chiji/gensyoku.html　がある）	
15	テーブルの組み合わせ検索	郵便番号コードから県知事をもとめるselect文を実行してみる 例: *SELECT DISTINCT(chiji.name) FROM ad_address, chiji WHERE ad_address.ken_id=chiji.ken_id AND ad_address.zip='001-0011';*	WHERE句の「=」で異なるテーブルの同一のキーをマッチさせる。 DISTINCTやad_address.ken_id=chiji.ken_id指定がないとどうなるかを試してみる
16	集計クエリ	住所.jpを集計する。都道府県別のレコード数。件数が多い順。 例: *SELECT ken_id, ken_name, count(*) FROM ad_address group by ken_id order by count(*) DESC;*	

5 . 図書館データを扱う

No	項目	作業内容	ポイント
17	複数テーブルの連動	book_all データベースを開く	
18	外部キー確認	[Tableプルダウン]と[Brows Dataタブ]で、書誌(books)と著者(maker_ncid)のテーブルを確認	テーブルの連携には外部キー（この例の場合は、NCID)を使う
18	テーブルの組み合わせ検索2	Execute SQLタブからNCIDをキーに複数テーブルを連動させる 　*SELECT books.*, maker_ncid.maker FROM books, maker_ncid WHERE books.ncid = maker_ncid.ncid and books.ncid ='BA66396868';*	
19	テーブルの組み合わせ検索3	*著者が「夏目, 漱石」の書誌レコードを表示する。* *select * from books where ncid in (select ncid from maker_ncid where maker='夏目, 漱石');*	SELECT文の「副問合せ」の例。WHERE句のSELECT文の結果をもとにさらに処理する。 INを指定することで、複数レコードの出力を可能に
20	他の実習用データ	実習用データ一式を紹介 　・NACSIS-CATオープンデータ全件　（「図書書誌」「雑誌書誌」「著者名典拠」「参加機関」） 　・CiNii Books図書書誌のISBD形式データサンプル100件 　・大学構成員ダミーデータ30件 　・図書受け入れダミーデータ40件 　・ILL複写依頼ダミーデータ10件	実際の図書館業務システムでの利用をイメージしているが、そのままでは不足。 実習でDBを作成するにあたっては、それを補うこと、また機能追加することを考えて欲しい。