2. SQL 問い合わせ（SQLite3, Python を使用）

データベース演習

URL: http://www.kkaneko.jp/cc/dbenshu/index.html

■ SQL 問い合わせ (SQL Query)

SQL 問い合わせは，リレーショナルデータベースに格納された 1 つまたは複数のテーブルを使う．例えば，次のように書くと，*m* 個のテーブル *T*1, *T*2, ..., *T*m の直積集合から条件 <*expression*> を満足する行のみを選び，そうして出来たテーブルの属性 *A*1, *A*2, ..., *A*n を出力するという意味になる．

```
SELDCT *A*1, *A*2, ..., *A*n
FROM    *T*1, *T*2, ,,,, *T*m
WHERE   <*expression*>
```

■ 条件 (Condition)

テーブル名とドット「.」と列名の並びを列の修飾名と呼ぶ．例えば，テーブル*R* の列 *A* の修飾名は *R.A* である．文字定数は，'X' のように，シングルクォーテーションマーク「'」で囲む．比較演算子は =, >, <, >=, <=, <> の 6 種類がある．探索条件は「R.A > 20」のように，列の修飾名と比較演算子と定数の並びである．扱うテーブルが 1 つのときは，列の修飾名の代わりに列名を使うことができる（「A > 20」のように）．条件は and や or で連結することができる．

Iris データセットのファイルは次のようなファイルである．

kaggle のデータセット： https://www.kaggle.com/datasets/uciml/iris から入手可能．

```
Id,SepalLengthCm,SepalWidthCm,PetalLengthCm,PetalWidthCm,Species
1,5.1,3.5,1.4,0.2,Iris-setosa
2,4.9,3.0,1.4,0.2,Iris-setosa
3,4.7,3.2,1.3,0.2,Iris-setosa
4,4.6,3.1,1.5,0.2,Iris-setosa
5,5.0,3.6,1.4,0.2,Iris-setosa
6,5.4,3.9,1.7,0.4,Iris-setosa
7,4.6,3.4,1.4,0.3,Iris-setosa
8,5.0,3.4,1.5,0.2,Iris-setosa
```

テーブル定義，テーブル生成，問い合わせ

① Python を起動する

```
python
```

② カレントディレクトリの確認

【プログラムの説明】

実行環境の確認のため，os モジュールを使用してカレントディレクトリを取得する．このパスはデータベースファイルと CSV ファイルの保存場所となり，後続の処理で重要な役割を果たす．

```
import os
os.getcwd()
```

③ テーブル定義を行う，次の Python プログラムを実行

データ型の指定は，各列で，integer, real などのデータ型を指定している．

id は主キーであるので「primary key」を指定している．

【プログラムの説明】

SQLite3 データベースの初期設定とテーブル作成を行う．create table 文で各列の型を指定し，id を主キーに設定．ヒアドキュメント形式で SQL を記述することで，複数行の SQL を見やすく表現している．

```
import pandas as pd
import sqlite3
c = sqlite3.connect('hoge.sqlite')
sql = u"""
create table iris (
   id integer          primary key,
   sepal_length      real,
   sepal_width       real,
   petal_length      real,
   petal_width       real,
   species              text );
"""
c.execute(sql)
```

④ テーブル生成（空のテーブルにレコードを挿入）を行う，次の Python プログラムを実行

プログラム中の「?」は SQL プレースホルダーである．

【プログラムの説明】

pandas（pd.read_csv）で CSV ファイルを読み込み，SQLite テーブルにデータを格納する．SQL プレースホルダー（?）と iterrows()を組み合わせることで，安全かつ効率的なデータ挿入を実現している．

(1) オブジェクト x に CSV ファイルを読み込む

```
x = pd.read_csv('c:/iris.csv', header=0)
```

Windows でのファイル名「C:¥iris.csv」は，Python のプログラム中では「'C:/iris.csv'」のように書く．

読み込みたい CSV ファイルの先頭行に，「id, sepal_length, sepal_width, petal_length, petal_width, species」のようなヘッダーがない場合には「header=None」を付ける．

（今回は，ヘッダーがあるので，「header=None」を付けない）

(2) オブジェクト x に格納されたデータを iris テーブルに挿入する

```
for index, r in x.iterrows():
    sql = u"insert into iris values (?, ?, ?, ?, ?, ?)"
    c.execute(sql, (r[0], r[1], r[2], r[3], r[4], r[5]))
```

「for r in x:」は x の各行について繰り返すという Python プログラム．

「insert into iris values」は，テーブルに 1 行挿入するという SQL プログラム．

「?」は，SQL プレースホルダー．

「r[0], r[1], r[2], r[3], r[4], r[5]」は，もとの CSV データファイルの 0 列目，1 列目，2 列目，3 列目，4 列目，5 列目を使うという意味．

```
c.commit()
```

⑤ テーブルをすべて読みだす．カーソルを使う．

【プログラムの説明】

カーソルを使用してテーブルの内容を確認する．SQL 問い合わせ（select * from iris）の結果をカーソルで取得し，for ループで表示．データベース操作終了後は close()で確実に接続を終了する．

```
cur = c.cursor()
cur.execute(u"select * from iris")
for t in cur:
    print (t)
```

「select * from iris」は，SQL 問い合わせ．

「cur」は，カーソルである．問い合わせ結果を得るのに使う．

テーブルから，条件に合致するレコードを得る．

```
cur = c.cursor()
cur.execute(u"select * from iris where id = 2")
```

```
for t in cur:
    print (t)

cur = c.cursor()
cur.execute(u" select * from iris where sepal_length > 7")
for t in cur:
    print (t)
```

⑦ より進んだ SQL 問い合わせの演習

【プログラムの説明】

特定の列の選択，複数条件の組み合わせなど，SQL の基本機能を使用した問い合わせを実行する．

```
# 特定の列のみを選択
cur.execute(u"select species, sepal_length, sepal_width from iris where id < 5")
print("特定の列を選択した結果：")
for t in cur:
    print (t)

# AND を使用した複数条件の組み合わせ
cur.execute(u"select * from iris where sepal_length > 5.0 and sepal_width > 3.5")
print("AND 条件を使用した結果：")
for t in cur:
    print (t)

# OR を使用した複数条件の組み合わせ
cur.execute(u"select id, species from iris where sepal_length < 4.5 or sepal_width > 4.0")
print("OR 条件を使用した結果：")
for t in cur:
    print (t)

c.close()
```

⑧ 終了処理として，データベース接続を切断する．

```
c.close()
```

⑨ 終了

```
exit()
```