```
Lab. Python Database Programming

1. SQLite3
  1)Python에서는 SQLite3에 대해 외부 module을 설치하지 않아도 된다.
  2)Python SQLite3 참조 문서
    https://goo.gl/8Ghx9r
  3)https://sqlite.org/index.html
    -Windows sqlite3
    -sqlite-tools-win32-x86-xxx.zip

  4)makedb.py
    import sqlite3

    conn = sqlite3.connect('myaddress.db')
    cursor = conn.cursor()

    cursor.execute('DROP TABLE IF EXISTS tblAddress')
    cursor.execute("""CREATE TABLE tblAddress
                      (name TEXT PRIMARY KEY, phone TEXT, addr TEXT)""")
    cursor.execute("INSERT INTO tblAddress VALUES ('한지민', '123-4567', '오산')")
    cursor.execute("INSERT INTO tblAddress VALUES ('이미자', '777-7777', '부산')")
    cursor.execute("INSERT INTO tblAddress VALUES ('설운도', '888-8888', '목포')")

    conn.commit()

    cursor.close()
    conn.close()

  5)selectdb.py
    import sqlite3

    conn = sqlite3.connect('myaddress.db')
    cursor = conn.cursor()

    cursor.execute('SELECT * FROM tblAddress')
    table = cursor.fetchall()
    for record in table:
        print('이름:%s, 전화:%s, 주소:%s' % record)

    cursor.close()
    conn.close()

  6)selectdb1.py
    import sqlite3

    conn = sqlite3.connect('myaddress.db')
    cursor = conn.cursor()

    cursor.execute('SELECT * FROM tblAddress')
    while True:
        record = cursor.fetchone()
```

```
        if record == None:
            break
        print('이름:%s, 전화:%s, 주소:%s' % record)

    cursor.close()
    conn.close()
```

7)DB Browser for SQLite
-https://sqlitebrowser.org/

8)sqlitedemo.py

```
import sqlite3

conn = sqlite3.connect('test.db')
cursor = conn.cursor()

cursor.execute("""CREATE TABLE IF NOT EXISTS supermarket
        (Itemno INTEGER, Category TEXT, FoodName TEXT,
        Company TEXT, Price INTEGER)""")

sql = "DELETE FROM supermarket"
cursor.execute(sql)

sql = """INSERT INTO supermarket(Itemno, Category, FoodName, Company, Price)
        VALUES(?,?,?,?,?)"""
cursor.execute(sql, (1, '과일', '자몽', '마트', 1500))

sql = """INSERT INTO supermarket(Itemno, Category, FoodName, Company, Price)
        VALUES(?,?,?,?,?)"""
cursor.execute(sql, (2, '음료수', '망고쥬스', '편의점', 1000))

conn.commit()

sql = """SELECT Itemno, Category, FoodName, Company, Price FROM
            supermarket"""
cursor.execute(sql)

while True:
    row = cursor.fetchone()
    if row == None:
        break
    print(str(row[0]) + " " + str(row[1]) + " " + str(row[2]) + " " + str(row[3]) + " " +
    str(row[4]))

cursor.close()
conn.close()
```

2. MySQL / MariaDB

1)cmd as Administrator
-If you’re used to using the “Run” box to open apps, you can use that to launch

Command Prompt with admin privileges.
-Press Windows+R to open the “Run” box.
-Type “cmd” into the box and then press Ctrl+Shift+Enter to run the command as an administrator.

2)pip install PyMySQL

3)mariadb.py

```
import pymysql

# Open database connection
db = pymysql.connect(host='localhost', port=3306, user='root', passwd='maria', db='estdb',charset='utf8',autocommit=True)

# prepare a cursor object using cursor() method
cursor = db.cursor()

# execute SQL query using execute() method.
cursor.execute("SELECT VERSION()")

# Fetch a single row using fetchone() method.
data = cursor.fetchone()
print ("Database version : %s " % data)

# disconnect from server
db.close()
```

4)Create Database mytest;
5)use mytest;
6)CREATE TABLE supermarket(

```
Itemno   INT   NULL,
Category  CHAR(20)  NULL,
FoodName  CHAR(30)  NULL,
Company   CHAR(20)   NULL,
Price INT  NULL);
```

7)INSERT INTO supermarket VALUES(1, '과일', '자몽', '마트', 1500)
INSERT INTO supermarket VALUES(2, '음료수', '망고주스', '편의점', 1000)
INSERT INTO supermarket VALUES(3, '음료수', '식혜', '시장', 1000)
INSERT INTO supermarket VALUES(4, '과자', '머랭', '조각케익가게', 3000)
8)SELECT * FROM supermarket;

9)mariadb1.py

```
import pymysql

server = 'localhost'
user = 'root'
password = 'pythonmariadb'
dbname = 'mytest'

conn = pymysql.connect(server, user, password, dbname, charset='utf8')
```

```
    cursor = conn.cursor()

    cursor.execute('SELECT * FROM supermarket;')

    row = cursor.fetchone()

    while row:
      print(str(row[0]) + " " + str(row[1]) + " " + str(row[2]) + " " + str(row[3]) + " " +
      str(row[4]))
      row = cursor.fetchone()

    conn.close()

  10)mariadb2.py
    import pymysql

    # Open database connection
    db = pymysql.connect("localhost","root","pythonmariadb","test" )

    # prepare a cursor object using cursor() method
    cursor = db.cursor()

    # Drop table if it already exist using execute() method.
    cursor.execute("DROP TABLE IF EXISTS EMPLOYEE")

    # Create table as per requirement
    sql = """CREATE TABLE EMPLOYEE (
      FIRST_NAME  CHAR(20) NOT NULL,
      LAST_NAME  CHAR(20),
      AGE INT,
      SEX CHAR(1),
      INCOME FLOAT )"""

    cursor.execute(sql)

    # disconnect from server
    db.close()

  11)mariadb3.py
    import pymysql

    # Open database connection
    db = pymysql.connect("localhost","root","pythonmariadb","test" )

    # prepare a cursor object using cursor() method
    cursor = db.cursor()

    # Prepare SQL query to INSERT a record into the database.
    sql = """INSERT INTO EMPLOYEE(FIRST_NAME,
      LAST_NAME, AGE, SEX, INCOME)
      VALUES ('Mac', 'Mohan', 20, 'M', 2000)"""
```

```
    try:
       # Execute the SQL command
       cursor.execute(sql)
       # Commit your changes in the database
       db.commit()
    except:
       # Rollback in case there is any error
       db.rollback()

    # disconnect from server
    db.close()

12)mariadb4.py
  import pymysql

    # Open database connection
    db = pymysql.connect("localhost","root","pythonmariadb","test" )

    # prepare a cursor object using cursor() method
    cursor = db.cursor()

    # Prepare SQL query to INSERT a record into the database.
    sql = "SELECT * FROM EMPLOYEE \
         WHERE INCOME > '%d'" % (1000)
    try:
       # Execute the SQL command
       cursor.execute(sql)
       # Fetch all the rows in a list of lists.
       results = cursor.fetchall()
       for row in results:
          fname = row[0]
          lname = row[1]
          age = row[2]
          sex = row[3]
          income = row[4]
          # Now print fetched result
          print ("fname = %s,lname = %s,age = %d,sex = %s,income = %d" % \
             (fname, lname, age, sex, income ))
    except:
       print ("Error: unable to fetch data")

    # disconnect from server
    db.close()

13)mariadb5.py
  import pymysql

    # Open database connection
    db = pymysql.connect("localhost","root","pythonmariadb","test" )

    # prepare a cursor object using cursor() method
```

```
    cursor = db.cursor()

    # Prepare SQL query to UPDATE required records
    sql = "UPDATE EMPLOYEE SET AGE = AGE + 1
                          WHERE SEX = '%c'" % ('M')
    try:
       # Execute the SQL command
       cursor.execute(sql)
       # Commit your changes in the database
       db.commit()
    except:
       # Rollback in case there is any error
       db.rollback()

    # disconnect from server
    db.close()

  14)mariadb6.py
    import pymysql

    # Open database connection
    db = pymysql.connect("localhost","root","pythonmariadb","test" )

    # prepare a cursor object using cursor() method
    cursor = db.cursor()

    # Prepare SQL query to DELETE required records
    sql = "DELETE FROM EMPLOYEE WHERE AGE > '%d'" % (20)
    try:
       # Execute the SQL command
       cursor.execute(sql)
       # Commit your changes in the database
       db.commit()
    except:
       # Rollback in case there is any error
       db.rollback()

    # disconnect from server
    db.close()


3. MariaDB World database 이용하기
  1)World database 다운로드하기
    -https://dev.mysql.com/doc/index-other.html
    -Example Databases에서 [World database] 'Zip' link 클릭
    -다운로드 후 C:/temp에 압축을 풀면 world.sql 파일이 보인다.

  2)MariaDB login한다.
    mysql -h localhost -u root -p

  3)C:/temp의 world.sql을 실행한다.
```

```
302     MariaDB [(none)]>source C:/temp/world.sql
303
304   4)World database의 table을 확인한다.
305     MariaDB [world]> show tables;
306
307   5)mariadb.py
308     import pymysql
309
310     # Open database connection
311     db = pymysql.connect("localhost","root","pythonmariadb","world" )
312
313     # prepare a cursor object using cursor() method
314     cursor = db.cursor()
315
316     sql = "SELECT ID, Name, CountryCode, District, Population FROM city WHERE
        CountryCode='KOR'"
317
318     try:
319       # Execute the SQL command
320       cursor.execute(sql)
321       # Fetch all the rows in a list of lists.
322       results = cursor.fetchall()
323       for row in results:
324         print('ID = %d, Name = %s, CountryCode = %s, District = %s, Popluation = %d' %
          (row[0], row[1], row[2], row[3],row[4]))
325     except:
326       print ("Error: unable to fetch data")
327
328     # disconnect from server
329     db.close()
330
331
332 4. mysql-connector-python module 이용하기
333   1)$ install mysql-connector-python
334
335     import mysql.connector as mariadb
336     from mysql.connector import Error
337
338     try:
339       connection = mariadb.connect(user='root', password='pythonmariadb',
          host='localhost', database='world')
340       if connection.is_connected() :
341         db_info = connection.get_server_info()
342         print('Connected to MariaDB database. MariaDB Server info is', db_info)
343         cursor = connection.cursor()
344         cursor.execute("SELECT ID, Name, CountryCode, District, Population FROM city
          WHERE CountryCode='KOR'")
345         for ID,Name,CountryCode,District,Population in cursor:
346           print('ID = %d, Name = %s, CountryCode = %s, District = %s, Popluation =
            %d' % (ID, Name, CountryCode,District,Population))
347     except Error as e :
```

```
        print('Error while connecting to MariaDB :', e)
    finally :
        #Closing database connection.
        if connection.is_connected() :
            cursor.close()
            connection.close()
            print("MariaDB connection is closed.")


5. Oracle
  1)Oracle Database 11g Express Edition Installation
  2)OracleServiceXE & OracleXETNSListener service start
  3)HR Schema activation
    -ALTER USER hr ACCOUNT UNLOCK IDENTIFIED BY hr;

  4)Oracle cx_Oracle 7.2
  5)cx_Oracle installation
    $ pip install cx_oracle --upgrade

    -In Anaconda Prompt
      $ conda install cx_oracle

  6)oracledb.py
    import cx_Oracle

    conn = cx_Oracle.connect('hr', 'hr', 'localhost:1521/XE')
    print(conn)   #<cx_Oracle.Connection to hr@localhost:1521/XE>

    conn1 = cx_Oracle.connect('hr/hr@localhost:1521/XE')
    print(conn1)   #<cx_Oracle.Connection to hr@localhost:1521/XE>

    dsn_tns = cx_Oracle.makedsn('localhost', 1521, 'XE')
    print(dsn_tns)
    #(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=localhost)(PORT=1521))(CONNE
    CT_DATA=(SID=XE)))

    conn2 = cx_Oracle.connect('hr', 'hr', dsn_tns)
    print(conn2)
    #<cx_Oracle.Connection to
    hr@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=localhost)(PORT=1521))(CON
    NECT_DATA=(SID=XE)))>

    print(conn.version)   #11.2.0.2.0

    #Cursor Objects
    conn3 = cx_Oracle.connect('hr', 'hr', 'localhost:1521/XE')
    cursor = conn3.cursor()
    sql = """SELECT employee_id, first_name, salary, to_char(hire_date, 'yyyy-mm-dd'),
    department_name, city
            from employees e inner join departments d on e.department_id = d.department_id
            inner join locations l on d.location_id = l.location_id"""
```

```
    cursor.execute(sql)

    for employee_id, first_name, salary, hire_date, department_name, city in cursor :
        print(employee_id, first_name, salary, hire_date, department_name, city)

    cursor.close()


6. MongoDB
  1)test 데이터베이스에 supermarket Collection 만들기
    -Command 창에서 MongoDB Server로 접속
      $ mongo
      MongoDB shell version v4.2.0
      connecting to:
      mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
      Implicit session: session { "id" : UUID("b8f7892c-63e2-4491-809b-11a108326b3f") }
      MongoDB server version: 4.2.0
      ...
      >

  2)test database로 연결
    > use test
    switched to db test

    > db    #현재 위치하고 있는 데이터베이스 확인
    test

    > show dbs    #모든 데이터베이스 보기
    admin   0.000GB
    config  0.000GB
    local   0.000GB

    > db.stats()    #현 test 데이터베이스 상태확인
    {
          "db" : "test",
          "collections" : 0,
          "views" : 0,
          "objects" : 0,
          "avgObjSize" : 0,
          "dataSize" : 0,
          "storageSize" : 0,
          "numExtents" : 0,
          "indexes" : 0,
          "indexSize" : 0,
          "scaleFactor" : 1,
          "fileSize" : 0,
          "fsUsedSize" : 0,
          "fsTotalSize" : 0,
          "ok" : 1
    }
```

```
3)supermarket Collection 생성하기
  > db.createCollection('supermarket')
  { "ok" : 1 }

  > show collections    #test 데이터베이스의 모든 collection 보기
  supermarket

4)데이터 insert

  > db.supermarket.insert([
  ... {
  ... Itemno:1,
  ... Category:'과일',
  ... FoodName:'자몽',
  ... Company:'마트',
  ... Price:1500
  ... },
  ... {
  ... Itemno:2,
  ... Category:'음료수',
  ... FoodName:'망고주스',
  ... Company:'편의점',
  ... Price:1000
  ... },
  ... {
  ... Itemno:3,
  ... Category:'음료수',
  ... FoodName:'식혜',
  ... Company:'시장',
  ... Price:1000
  ... },
  ... {
  ... Itemno:4,
  ... Category:'과자',
  ... FoodName:'머랭',
  ... Company:'조각케잌가게',
  ... Price:3000
  ... }
  ... ])
  BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 4,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
  })

5)지금까지 insert한 데이터 모두 select하기
```

```
> db.supermarket.find().pretty()
{
        "_id" : ObjectId("5d5e3470c8076c145c95fb6f"),
        "Itemno" : 1,
        "Category" : "과일",
        "FoodName" : "자몽",
        "Company" : "마트",
        "Price" : 1500
}
{
        "_id" : ObjectId("5d5e3470c8076c145c95fb70"),
        "Itemno" : 2,
        "Category" : "음료수",
        "FoodName" : "망고주스",
        "Company" : "편의점",
        "Price" : 1000
}
{
        "_id" : ObjectId("5d5e3470c8076c145c95fb71"),
        "Itemno" : 3,
        "Category" : "음료수",
        "FoodName" : "식혜",
        "Company" : "시장",
        "Price" : 1000
}
{
        "_id" : ObjectId("5d5e3470c8076c145c95fb72"),
        "Itemno" : 4,
        "Category" : "과자",
        "FoodName" : "머랭",
        "Company" : "조각케잌가게",
        "Price" : 3000
}


6)mongodbdemo.py
  import json
  import pymongo
  from pymongo import MongoClient

  #Database connection
  client = MongoClient('localhost', 27017, maxPoolSize = 50)

  #supermarket collection finding
  db = client.test
  collection = db['supermarket']
  cursor = collection.find({}, {'_id':False})

  #cursor로 looping(i에는 0부터 loop의 숫자가 들어간다)
  for i, document in enumerate(cursor):
```

```
547             rowcontent = ''
548             keycontent = ''
549             #print(i, document)
550 
551             # #document에서 key와 값을 가져와서 tab으로 구분해서 문자열로 만듦
552             for key, val in document.items():
553                 # print(key, val)
554                 keycontent = keycontent + '\t' + str(key)
555                 rowcontent = rowcontent + '\t' + str(val)
556 
557             #만약 첫번째 loop라면 key 이름도 출력해서 column 이름 출력한다.
558             if i == 0 :
559                 print(keycontent)
560                 print(rowcontent)
561             else:
562                 print(rowcontent)
563             i += 1
```