

윈도우 프로그래밍

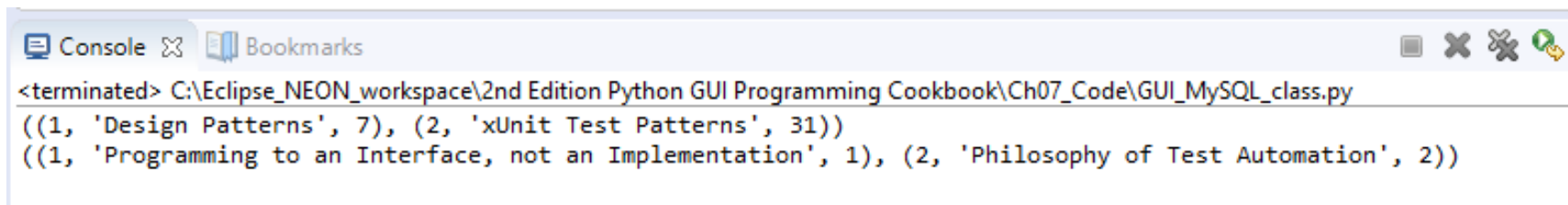
세명대학교 컴퓨터학부
담당교수: 송유정



SQL UPDATE 명령 사용하기

- MySQL Class 코드에서 아래 코드를 작성해 실행하면 현재 books와 quotations 테이블에 있는 데이터 확인 가능

```
#=====
if __name__ == '__main__':
    mySQL = MySQL()
    mySQL.showData()
```



```
<terminated> C:\Eclipse_NEON_workspace\2nd Edition Python GUI Programming Cookbook\Ch07_Code\GUI_MySQL_class.py
((1, 'Design Patterns', 7), (2, 'xUnit Test Patterns', 31))
((1, 'Programming to an Interface, not an Implementation', 1), (2, 'Philosophy of Test Automation', 2))
```

SQL UPDATE 명령 사용하기

- UPDATE 명령을 포함하는 함수 생성

```
def updateGOF_commit(self):  
    # connect to MySQL  
    conn, cursor = self.connect()  
    cursor = conn.cursor(buffered=True)
```

```
    self.useGuiDB(cursor)
```

```
    # execute command
```

```
    cursor.execute("SELECT Book_ID FROM books WHERE Book_Title = 'Design Patterns'")
```

```
    primKey = cursor.fetchall()[0][0]
```

```
    # print(primKey)
```



Book 테이블에서 Design patterns 책을 찾아 해당 데이터의 ID 조회, 첫번째 조회 결과를 primkey에 삽입

```
    cursor.execute("SELECT * FROM quotations WHERE Books_Book_ID = (%s)", (primKey,))
```

```
    # print(cursor.fetchall())
```



Quotation 테이블에서 위에서 조회한 ID 값에 해당하는 데이터 조회

```
    cursor.execute("UPDATE quotations SET Quotation = (%s) WHERE Books_Book_ID = (%s)", \  
                  ("Pythonic Duck Typing: If it walks like a duck and talks like a duck it probably is a duck...", primKey))
```

```
    # commit transaction
```

```
    conn.commit_()
```



Quotation 에서 조회된 데이터에 Quotation 컬럼 데이터를 "Pythonic~"으로 UPDATE하는 쿼리

```
    cursor.execute("SELECT * FROM quotations WHERE Books_Book_ID = (%s)", (primKey,))
```

```
    print(cursor.fetchall())
```

```
    # close cursor and connection
```

```
    self.close(cursor, conn)
```

SQL UPDATE 명령 사용하기

- 쿼리 결과를 Return해 확인하기 위한 함수 정의
- Book과 quotations 테이블의 모든 데이터 조회

```
def showDataWithReturn(self):  
    # connect to MySQL  
    conn, cursor = self.connect()  
  
    self.useGuiDB(cursor)  
  
    # execute command  
    cursor.execute("SELECT * FROM books")  
    booksData = cursor.fetchall()  
  
    cursor.execute("SELECT * FROM quotations")  
    quoteData = cursor.fetchall()  
  
    # close cursor and connection  
    self.close(cursor, conn)  
  
    # print(booksData, quoteData)  
    for record in quoteData:  
        print(record)  
  
    return booksData, quoteData
```

```
(1, 'Pythonic Duck Typing: If it walks like a duck and talks like a duck it probably is a duck...', 1)  
(2, 'Programming to an Interface, not an Implementation', 2)  
(3, 'Philosophy of Test Automation', 3)  
(4, '\n', 4)
```

```
if __name__ == '__main__':  
    mySQL = MySQL()  
  
    mySQL.updateGOF_commit()  
    book, quote = mySQL.showDataWithReturn()  
    print(book, quote)
```

SQL UPDATE 명령 사용하기

- 실행 결과 확인하기 -> Design patterns 제목을 갖는 책 인용문 변화 확인
- Select * from 구문 실행

```
mysql> USE guidb
Database changed
mysql> SELECT * FROM books;
+-----+-----+-----+
| Book_ID | Book_Title          | Book_Page |
+-----+-----+-----+
|      1 | Design Patterns     |          7 |
|      2 | xUnit Test Patterns |         31 |
+-----+-----+-----+
2 rows in set (0.10 sec)

mysql> SELECT * FROM quotations;
+-----+-----+-----+
| Quote_ID | Quotation                                                    | Books_Book_ID |
+-----+-----+-----+
|      1 | Programming to an Interface, not an Implementation         |              1 |
|      2 | Philosophy of Test Automation                               |              2 |
+-----+-----+-----+
```

```
mysql> SELECT * FROM books;
+-----+-----+-----+
| Book_ID | Book_Title          | Book_Page |
+-----+-----+-----+
|      1 | Design Patterns     |          7 |
|      2 | xUnit Test Patterns |         31 |
+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> SELECT * FROM quotations;
+-----+-----+-----+
| Quote_ID | Quotation                                                    | Books_Book_ID |
+-----+-----+-----+
|      1 | Pythonic Duck Typing: If it walks like a duck and talks like a duck it probably is a duck... |              1 |
|      2 | Philosophy of Test Automation                               |              2 |
+-----+-----+-----+
```

SQL Delete 명령 사용하기

- UPDATE에서 생성한 데이터 삭제하기
- 우리가 생성한 2개의 테이블은 기본키와 외래 키 관계로 구성되어 있기 때문에, 다중 삭제 처리 등을 설정해야 함
- 고아 레코드 확인하기 (test)
- 외래 키 관계가 없는 테이블을 만들기
- createTablesNoFK 함수를 활용하여 새로운 테이블 생성

```
def createTablesNoFK(self):  
    # connect to MySQL  
    conn, cursor = self.connect()  
  
    self.useGuiDB(cursor)  
  
    # create Table inside DB  
    cursor.execute("CREATE TABLE nBooks ( \n  
        Book_ID INT NOT NULL AUTO_INCREMENT, \n  
        Book_Title VARCHAR(25) NOT NULL, \n  
        Book_Page INT NOT NULL, \n  
        PRIMARY KEY (Book_ID) \n  
    ) ENGINE=InnoDB")  
  
    # create second Table inside DB --  
    # No FOREIGN KEY relation to Books Table  
    cursor.execute("CREATE TABLE nQuotations ( \n  
        Quote_ID INT AUTO_INCREMENT, \n  
        Quotation VARCHAR(250), \n  
        Books_Book_ID INT, \n  
        PRIMARY KEY (Quote_ID) \n  
    ) ENGINE=InnoDB")  
  
    # close cursor and connection  
    self.close(cursor, conn)
```

SQL Delete 명령 사용하기

- Main에서 table 생성 및 예제 데이터 삽입

```
if __name__ == '__main__':  
    mySQL = MySQL()  
    mySQL.createTablesNoFK()  
    mySQL.insertnBooksExample()  
    mySQL.insertnBooksExample()  
    mySQL.insertnBooksExample() #nbooks에 삽입되도록 코드 수정
```



```
if __name__ == '__main__':  
    mySQL = MySQL()  
    #mySQL.createTablesNoFK()  
    #mySQL.insertnBooksExample()  
    #mySQL.insertnBooksExample()  
    #mySQL.insertnBooksExample() #nbooks에 삽입되도록 코드 수정  
    mySQL.updateGOF_commit() #nbooks에 삽입되도록 코드 수정, 데이터 업데이트
```

SQL Delete 명령 사용하기

- UPDATE에서 생성한 데이터 삭제하기

```
if __name__ == '__main__':  
    mySQL = MySQL()  
    #mySQL.createTablesNoFK()  
    #mySQL.insertnBooksExample()  
    #mySQL.insertnBooksExample()  
    #mySQL.insertnBooksExample() #nbooks에 삽입되도록 코드 수정  
    #mySQL.updateGOF_commit() #nbooks에 삽입되도록 코드 수정, 데이터 업데이트  
    mySQL.deleteRecord()
```

```
MySQL localhost:3306 ssl guidb SQL > SELECT * FROM nBOOKS;
```

Book_ID	Book_Title	Book_Page
2	Design Patterns	17
3	Design Patterns	17
4	Design Patterns	17
5	Design Patterns	17

```
4 rows in set (0.0004 sec)
```

```
MySQL localhost:3306 ssl guidb SQL > SELECT * FROM NQuotations;
```

Quote_ID	Quotation	Books_Book_ID
1	Pythonic Duck Typing: If it walks like a duck and talks like a duck it probably is a duck...	1
2	Programming to an Interface, not an Implementation	2
3	Programming to an Interface, not an Implementation	3
4	Programming to an Interface, not an Implementation	4
5	Programming to an Interface, not an Implementation	5

SQL Delete 명령 사용하기

- 앞선 상황은 데이터 손상의 원인이 되며, CASCADE를 이용해 피할 수 있음

```
cursor.execute("CREATE TABLE Quotations ( \n\n    Quote_ID INT AUTO_INCREMENT, \n    Quotation VARCHAR(250), \n    Books_Book_ID INT, \n    PRIMARY KEY (Quote_ID), \n    FOREIGN KEY (Books_Book_ID) \n        REFERENCES Books(Book_ID) \n        ON DELETE CASCADE \n\n) ENGINE=InnoDB")
```

```
MySQL localhost:3306 ssl guidb SQL > SELECT * FROM BOOKS;
```

Book_ID	Book_Title	Book_Page
2	Design Patterns	7
3	xUnit Test Patterns	31
4	오만과 편견	12
5	오만과 편견	23
9	1123	123
10	HIHELLO	222
11	test	112
12	book sample	52
13	book sample	52

```
9 rows in set (0.0003 sec)\nMySQL localhost:3306 ssl guidb SQL > SELECT * FROM Quotations;
```

Quote_ID	Quotation
2	Pythonic Duck Typing: If it walks like a duck and talks like a duck it probably is a duck...
3	Philosophy of Test Automation
4	
5	{1 {Design Patterns} 17} {2 {Design Patterns} 7} {3 {xUnit Test Patterns} 31} {4 {오만과 편견} 5}
6	
7	{1 {Design Patterns} 17} {2 {Design Patterns} 7} {3 {xUnit Test Patterns} 31} {4 {오만과 편견} 10}
8	
9	quotation sample
10	quotation sample

MySQL 워크벤치 사용하기

- Workbench 사용하기

The screenshot displays the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. Below the menu is a toolbar with various icons. The main window is titled 'Query 1' and contains the following SQL queries:

```
2 • SELECT * FROM books;
3 • SELECT * FROM quotations;
```

Below the query editor, the 'Result Grid' tab is active, showing the results of the queries. The results are displayed in a table with columns: Quote_ID, Quotation, and Books_Book_ID.

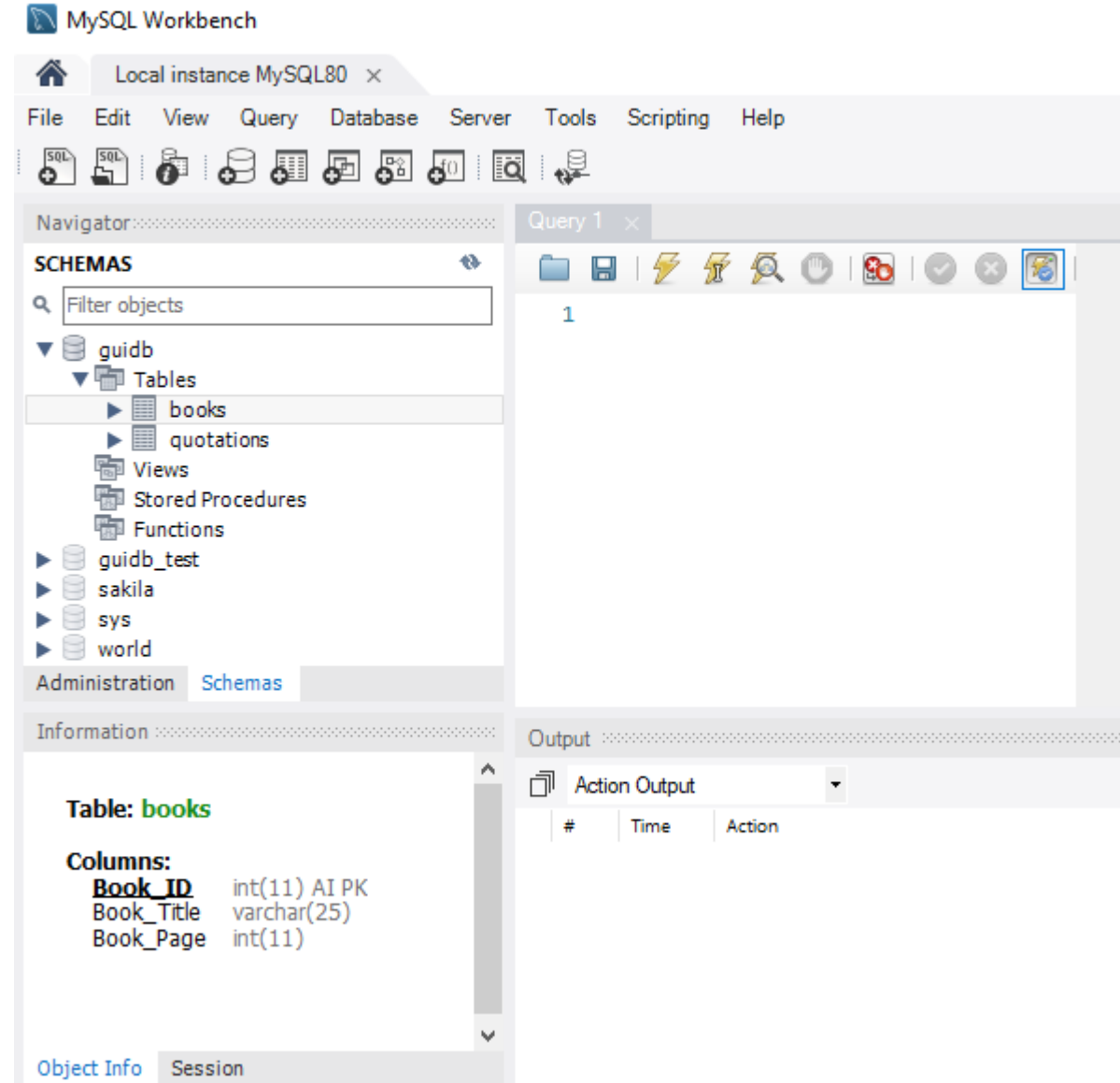
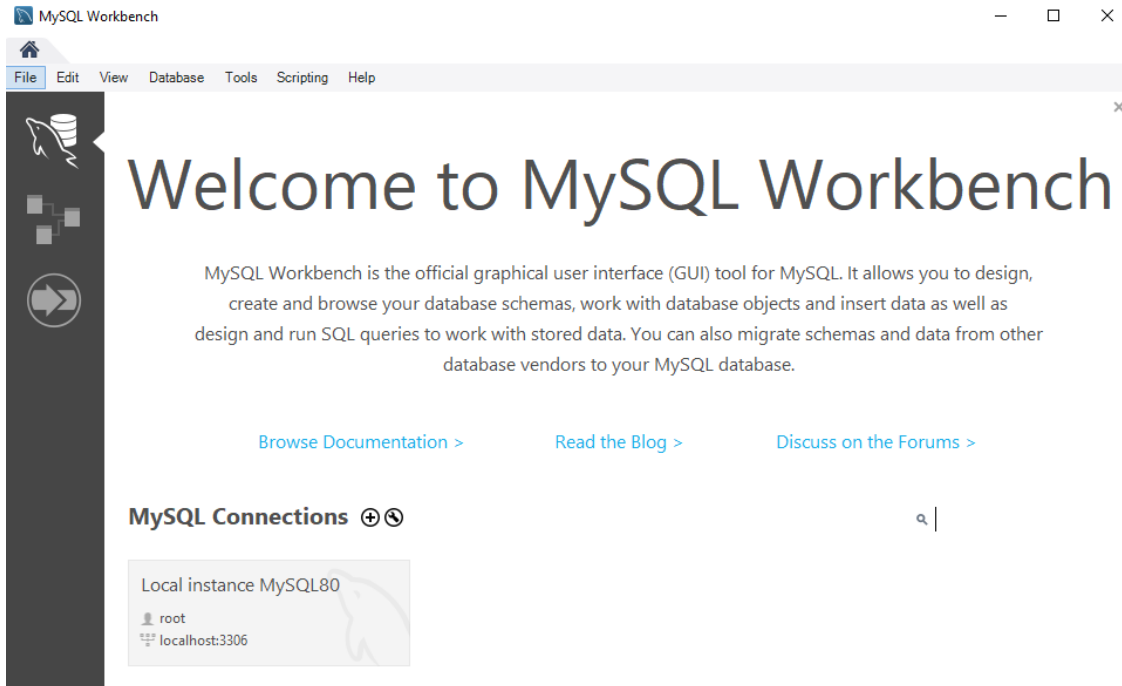
Quote_ID	Quotation	Books_Book_ID
1	Programming to an Interface, not an Implement...	1
2	Programming to an Interface, not an Implement...	2
3	Philosophy of Test Automation	3
NULL	NULL	NULL

At the bottom of the interface, the 'Output' tab is active, showing the 'Action Output' log. The log contains the following entries:

#	Time	Action	Message	Duration / Fetch
1	06:51:44	use guidb	0 row(s) affected	0.000 sec
2	06:51:44	SELECT * FROM books LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec
3	06:51:44	SELECT * FROM quotations LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec

MySQL 워크벤치 사용하기

- Workbench 사용하기



(실습) 데이터베이스 GUI 마무리

- Mody quote 버튼 동작 마무리
- 현재는 동작하지 않은 “mody quote” 버튼에 update 쿼리 동작 함수를 연결
- Book title과 변경할 quotations를 입력 시 데이터 갱신되도록 코드를 작성하시오.

The image shows a Python GUI application window titled "Python GUI" with a "MySQL" tab. The GUI has a "Python Database" section with three input fields for "Book Title:" and "Page:". The first field contains "오만과 편견" and the second is empty. There are three buttons: "Insert Quote", "Get Quotes", and "Mody Quote". Below these is a "Book Quotation" text area containing "바뀌어라 인용문". To the right, a snippet of SQL code is visible: `ERE Book_Title = (%s)'` and `ERE Books_Book_ID = (%s)`.

Below the GUI is a "MySQL Shell" terminal window. It displays a table of book data and the results of a SQL query.

Book_ID	Book_Title	Book_Page
2	Design Patterns	7
3	xUnit Test Patterns	31
4	오만과 편견	12
5	오만과 편견	23
9	1123	123
10	HIHELL0	222
11	test	112
12	book sample	52
13	book sample	52

9 rows in set (0.0004 sec)

```
MySQL localhost:3306 ssl guidb SQL> select *from quotations;
```

Quote_ID	Quotation
2	Pythonic Duck Typing: If it walks like a duck and talks like a du
3	Philosophy of Test Automation
4	<u>바뀌어라 인용문</u>
5	{1 {Design Patterns} 17} {2 {Design Patterns} 7} {3 {xUnit Test P
6	5
7	{1 {Design Patterns} 17} {2 {Design Patterns} 7} {3 {xUnit Test P
8	
9	quotation sample